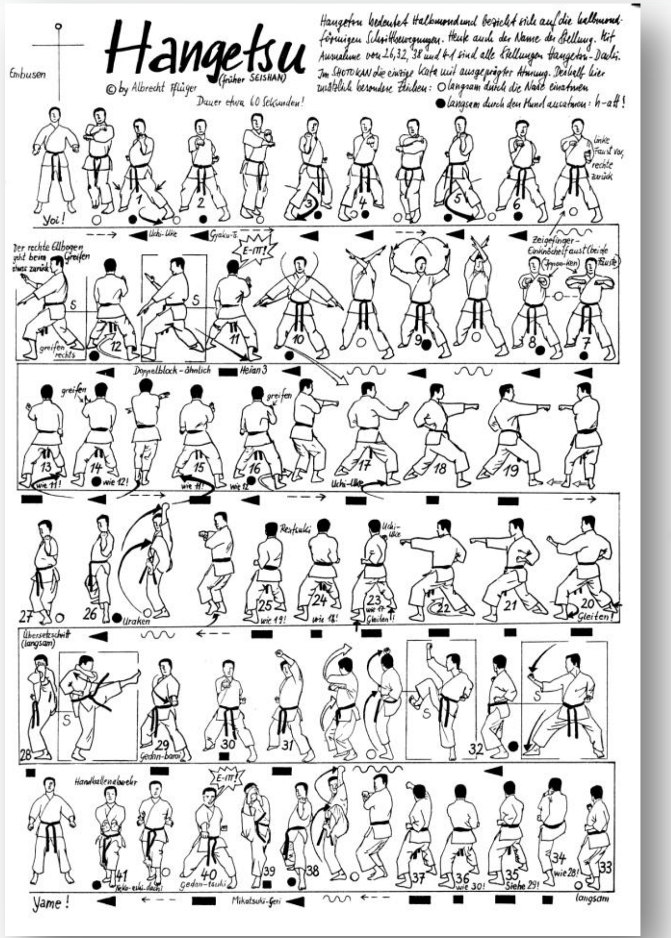
Practicing Advanced Unit Testing

with the «Trading Card Game» Kata

Battlecry: Draw a slide. **Deathrattle:** Transform audience into 1/1 sheep.

v2.3.2 (27.11.2014)

Kata



Hangetsu bedeutet Halbmond und bezieht sich auf die halbmondförmigen Schrittbewegungen. Heute auch der Name der Stellung. Mit Ausnahme von 26, 32, 38 und 41 sind alle Stellungen Hangetsu-Dachi. Im SHOTOKAN die einzige Kata mit ausgeprägter Atmung. Deshalb hier zusätzlich besondere Zeichen: ○ langsam durch die Nase einatmen ● langsam durch den Mund ausatmen: h-aa!

Kata (型 or 形 *literally: "form"*) is a Japanese word describing detailed choreographed patterns of movements practiced either solo or in pairs.

Trading Card Game (TCG)

A […] **trading card game** (**TCG**) […] is a card game that uses specially designed sets of playing cards […] mass-produced for trading or collectibility, and it must have rules for strategic game play. Acquiring these cards may be done by trading with other players or buying card packs.

Why invent another Kata?

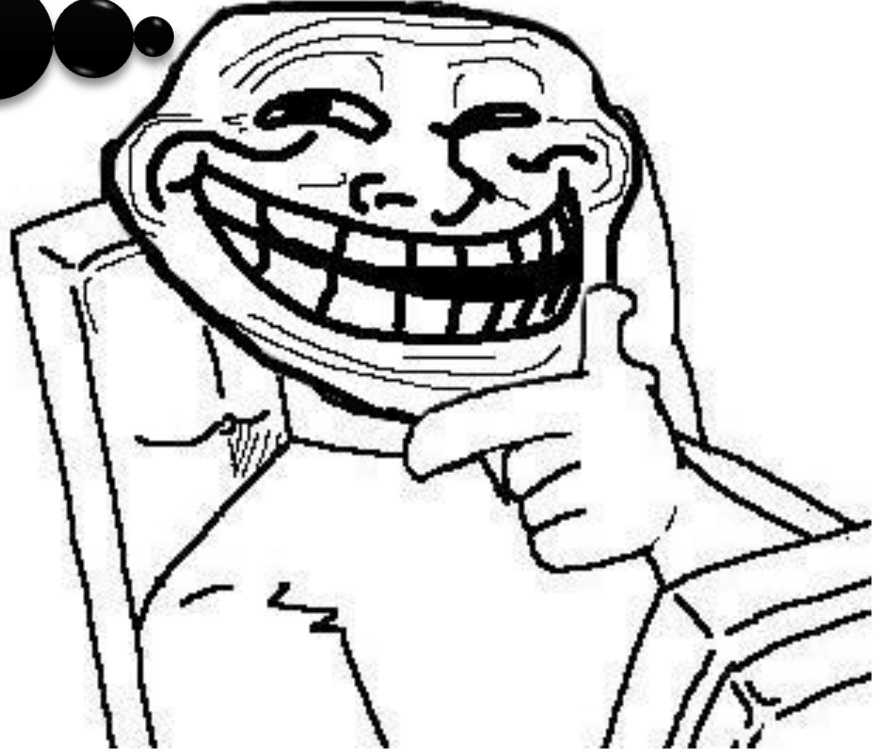
Bro, why U not use

?????

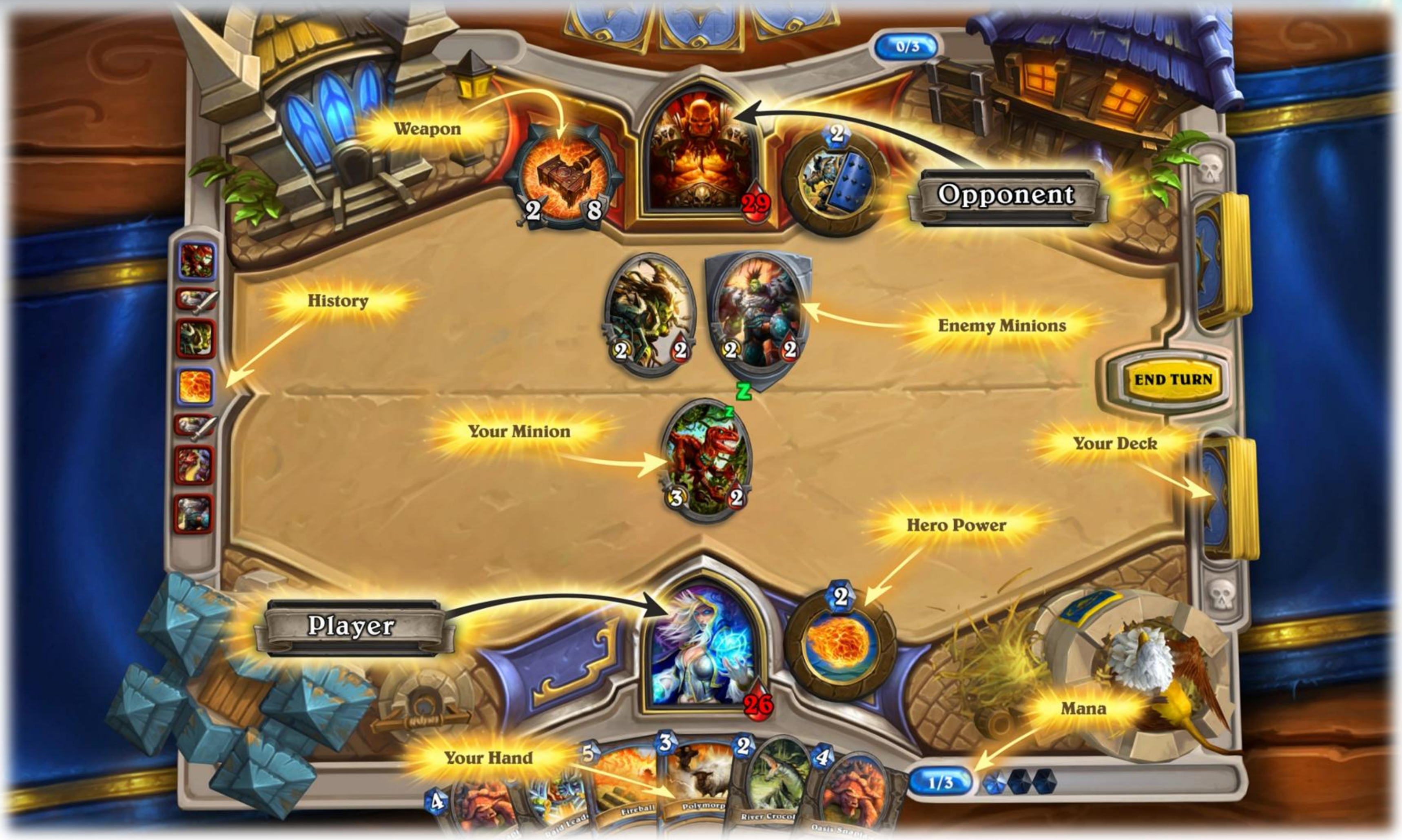
Not TDD-friendly

Too easy

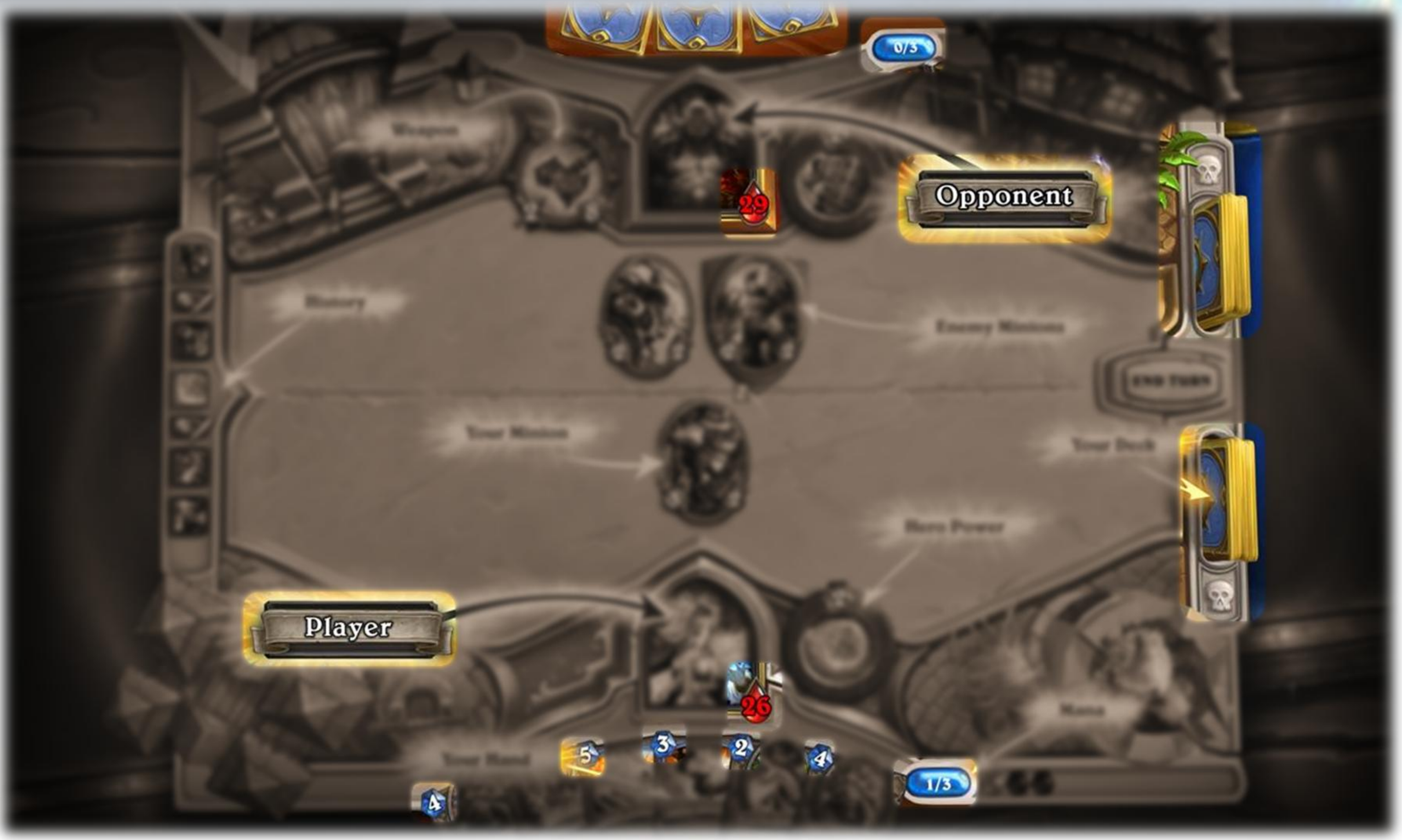
Rules not extensible



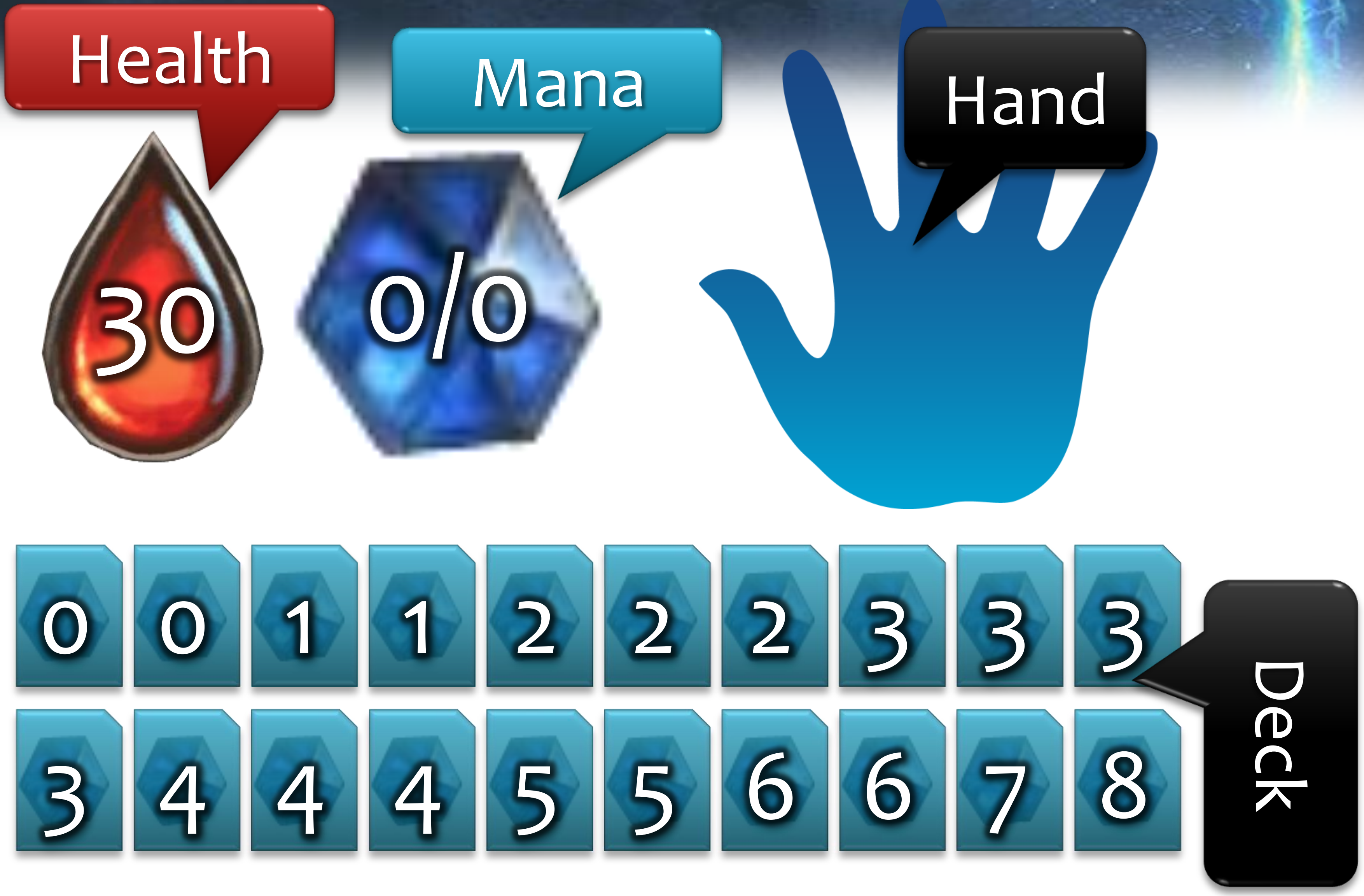
Hearthstone: Heroes of Warcraft®



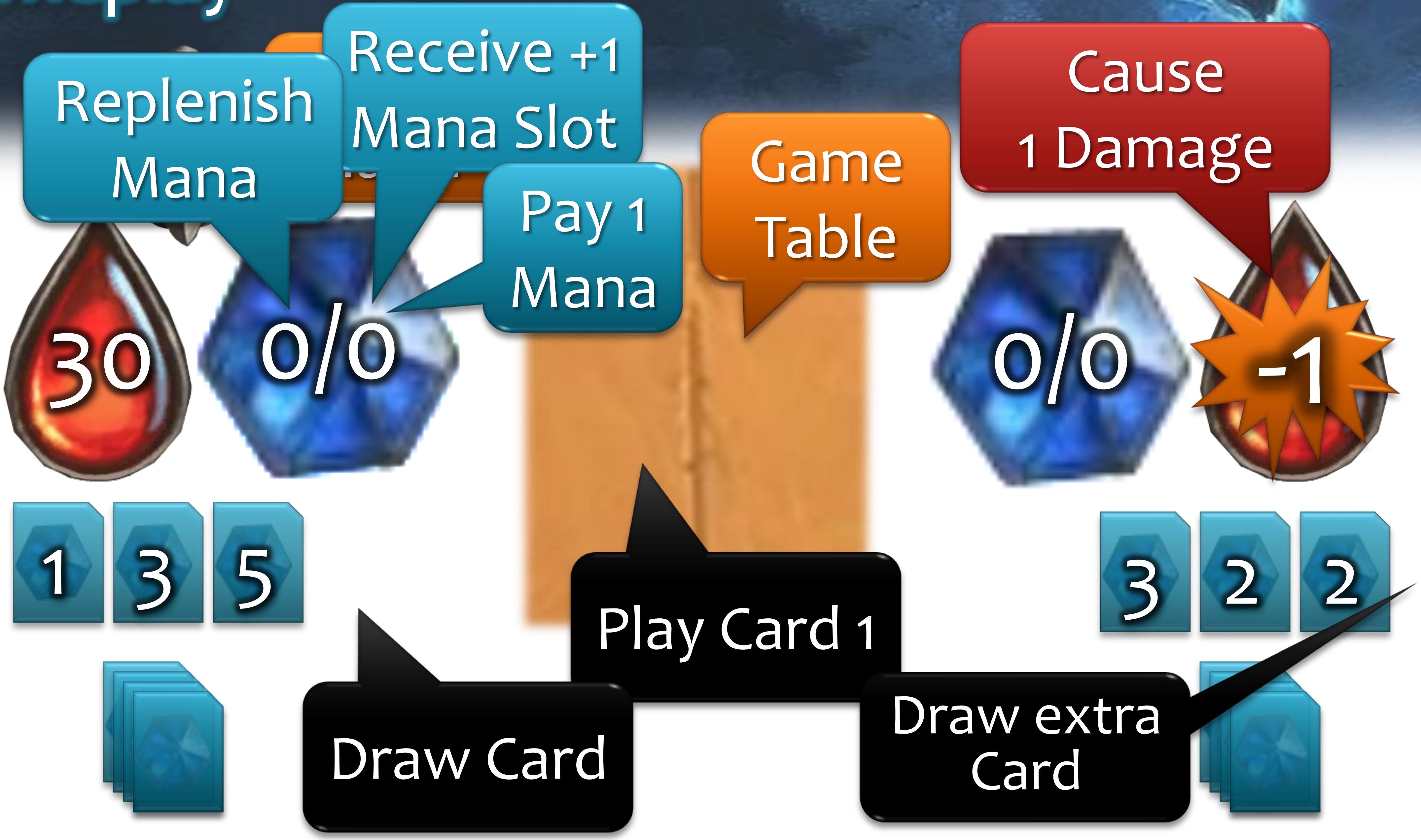
Kata Trading Card Game



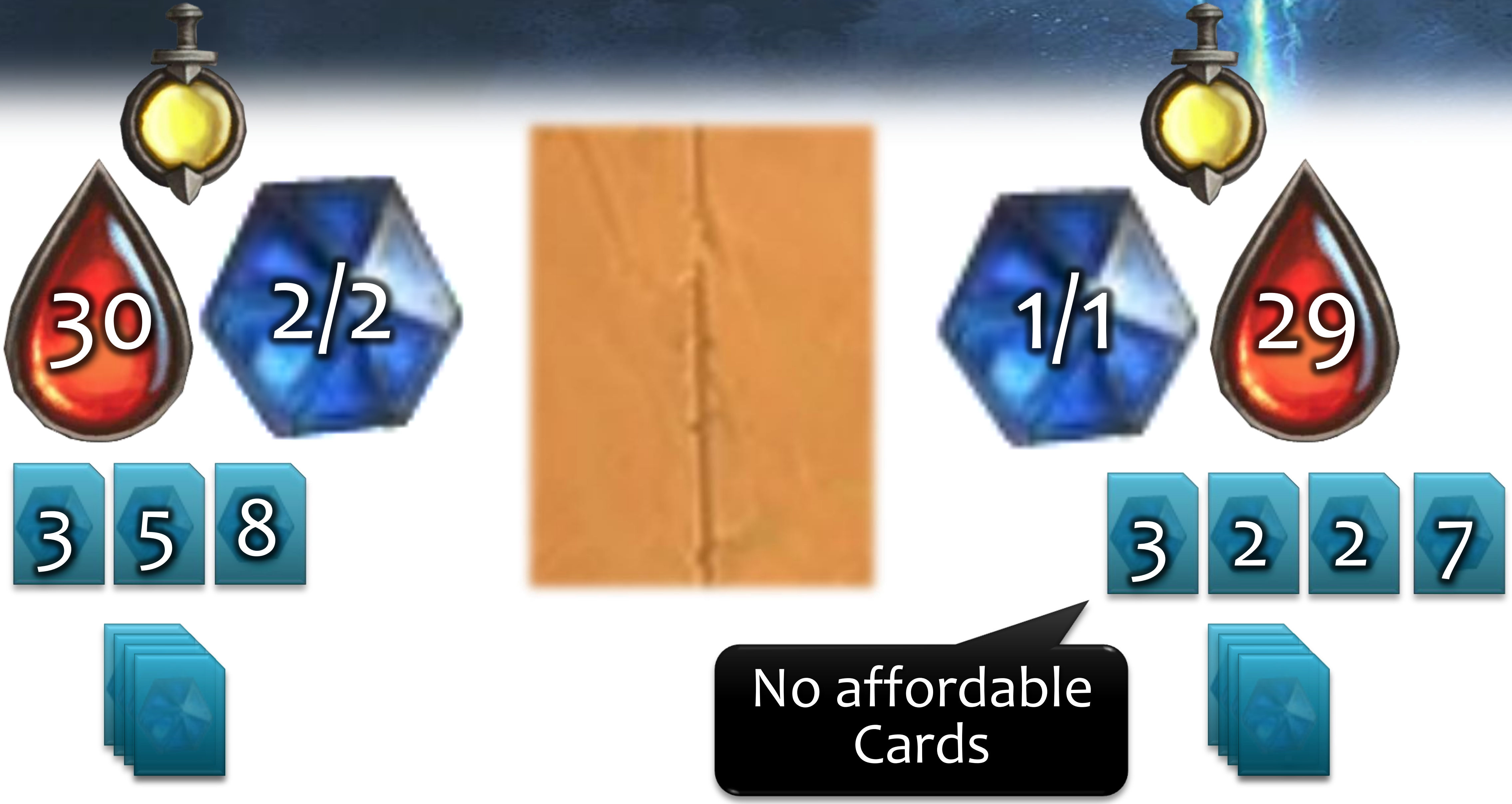
Player Setup



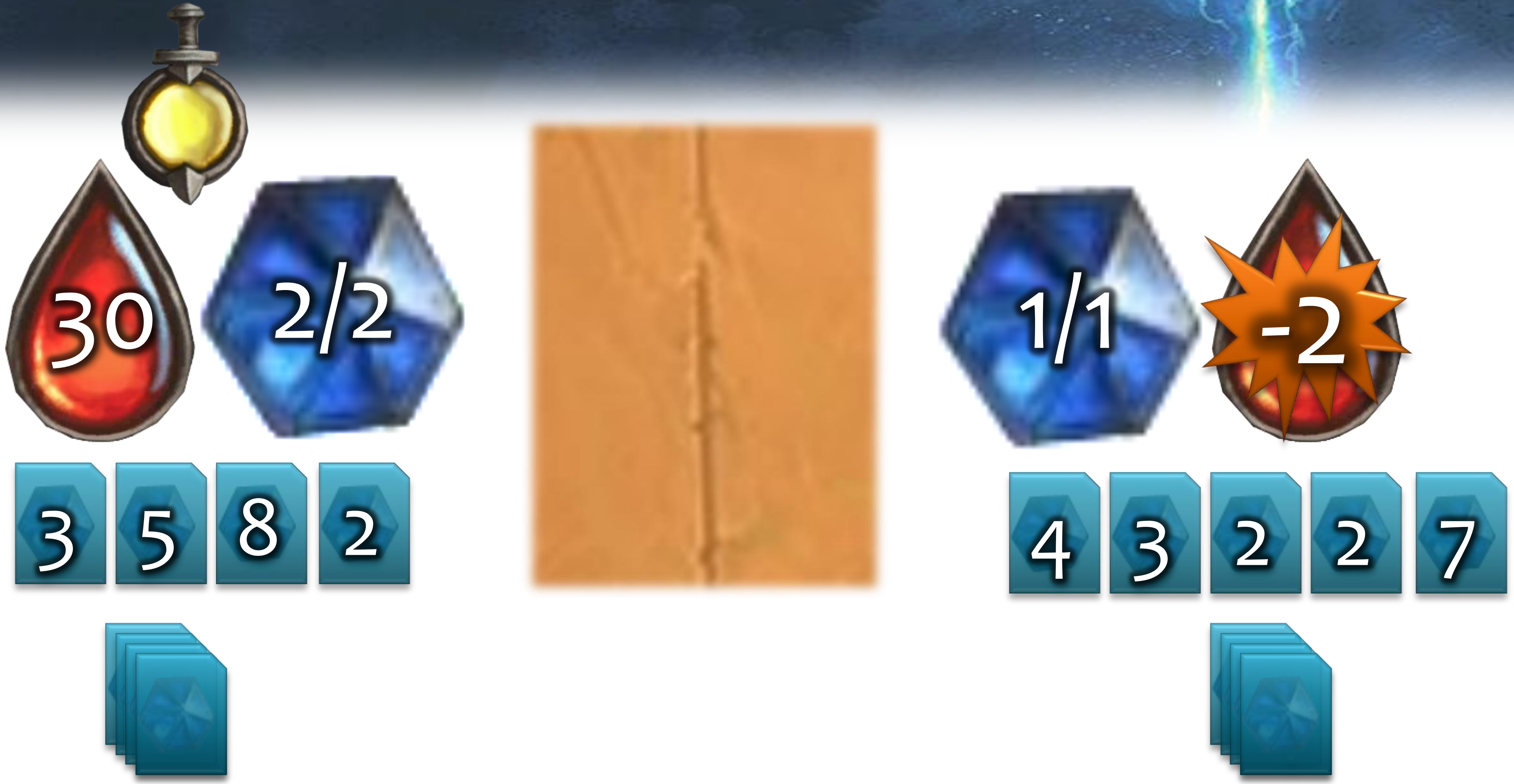
Gameplay


Replenish Mana
Receive +1 Mana Slot
Pay 1 Mana
Game Table
Cause 1 Damage
30
0/0
0/0
-1
1
3
5
3
2
2
Play Card 1
Draw Card
Draw extra Card

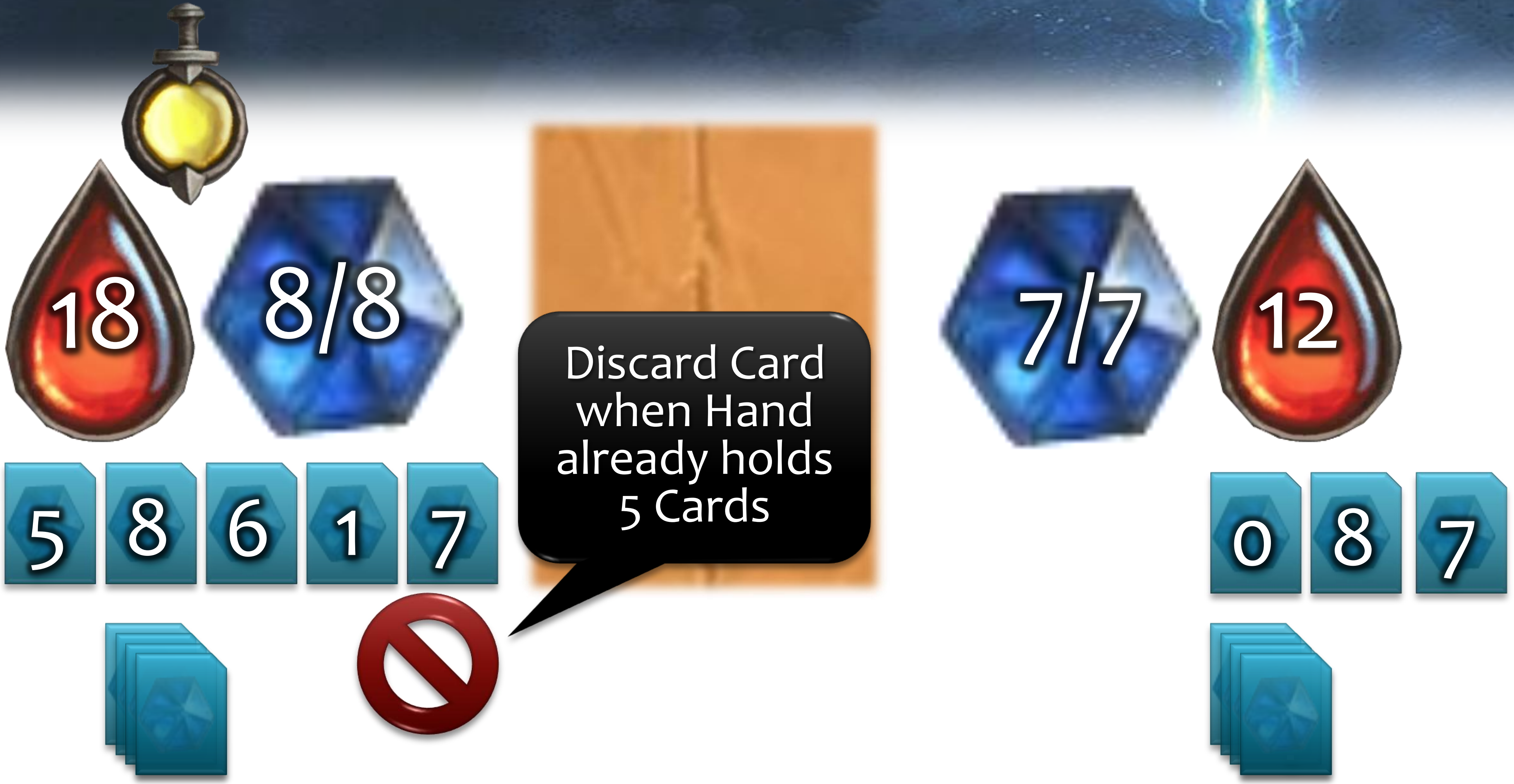
Forced Turn Skip



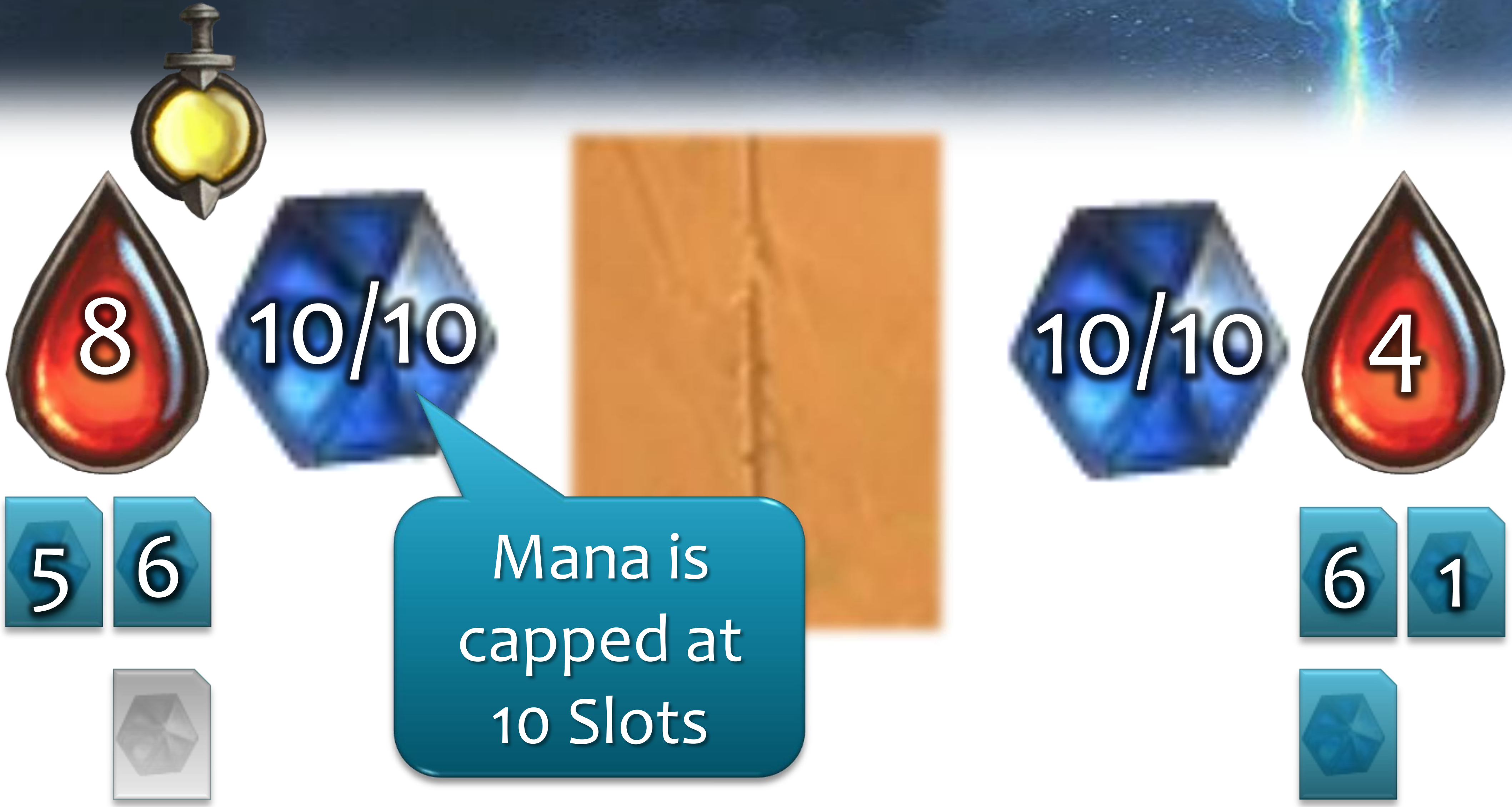
Ongoing Gameplay



Overload Rule

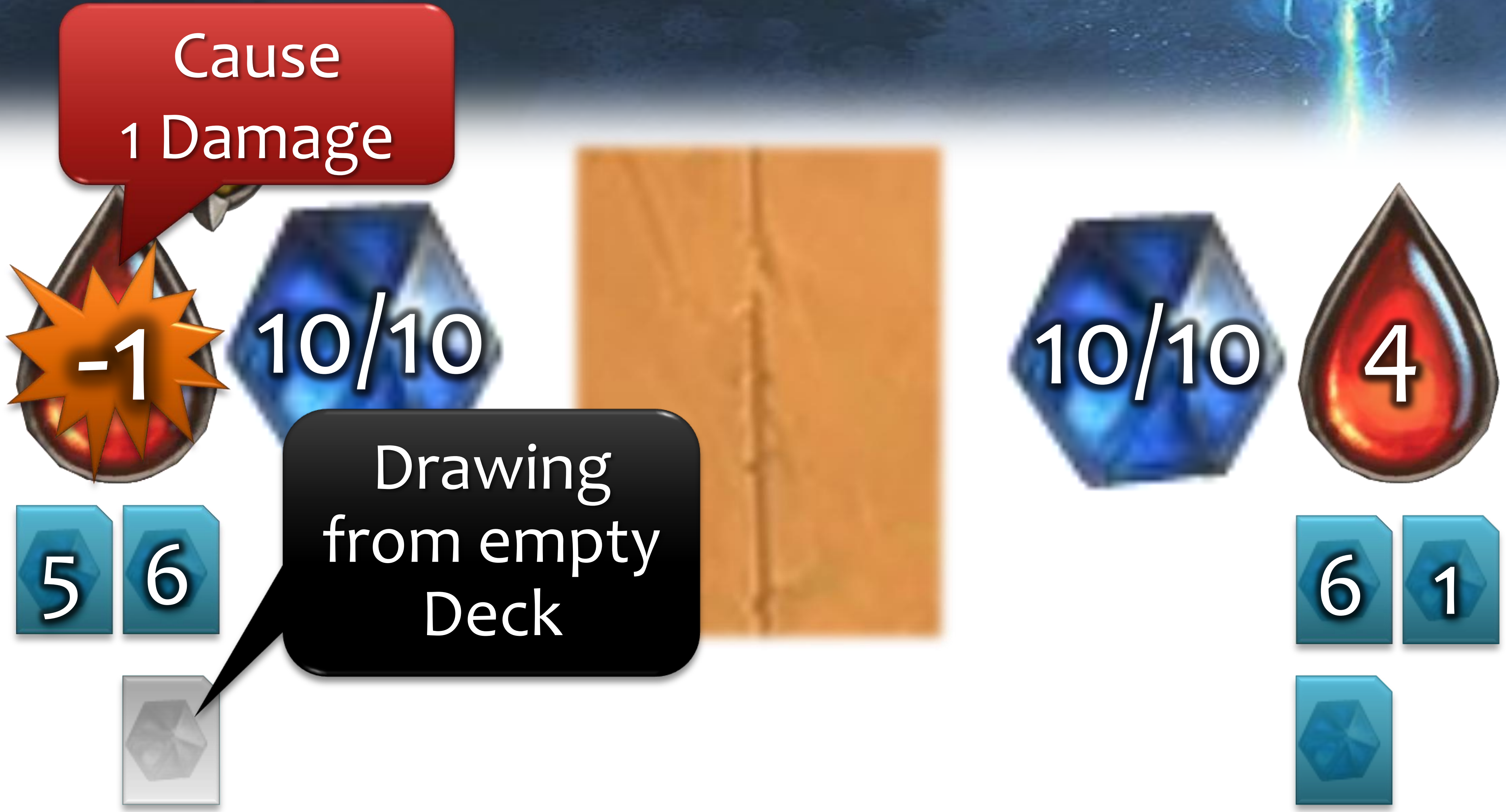


Mana Cap

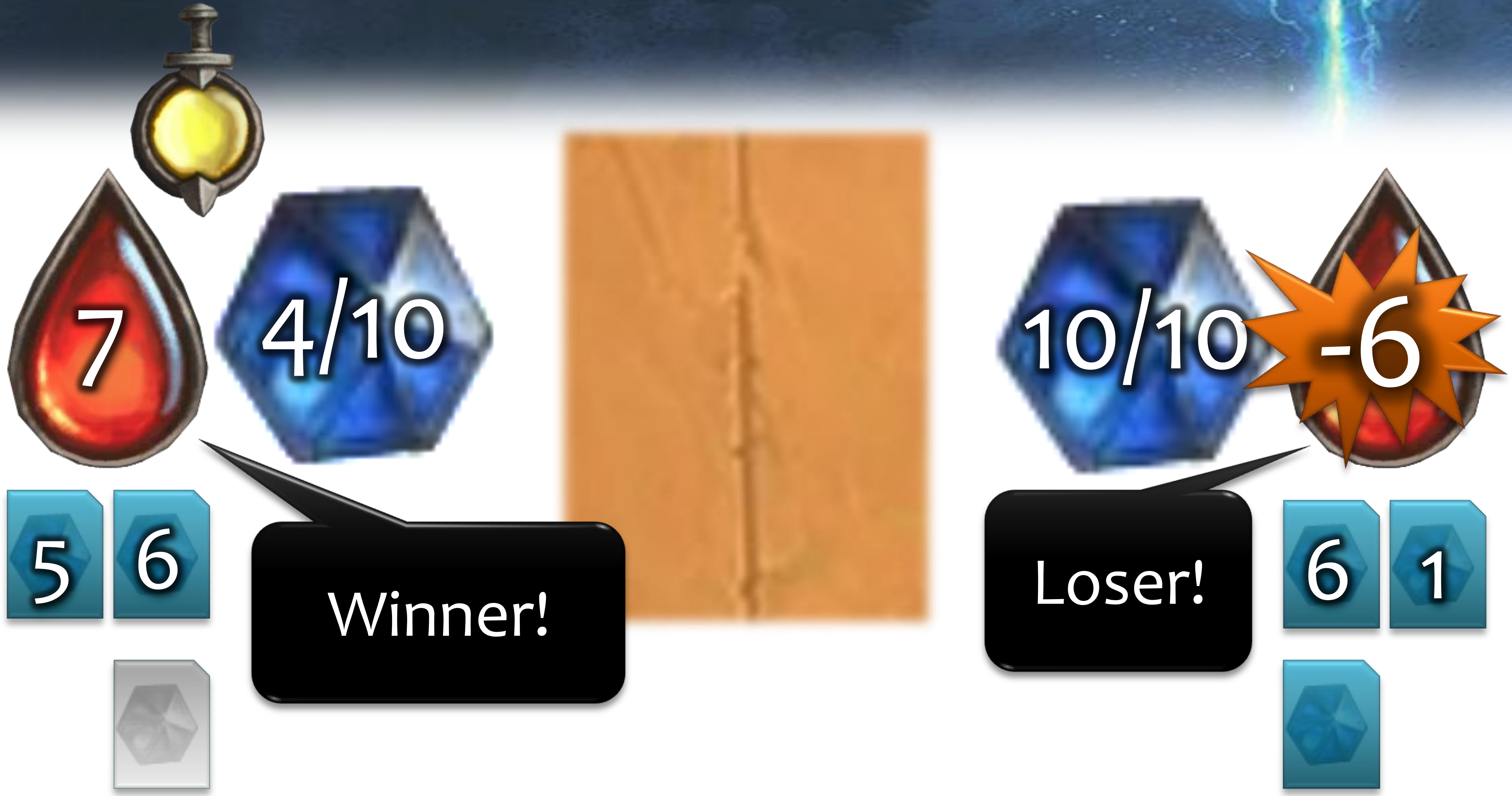


Mana is capped at 10 Slots

Bleedout Rule

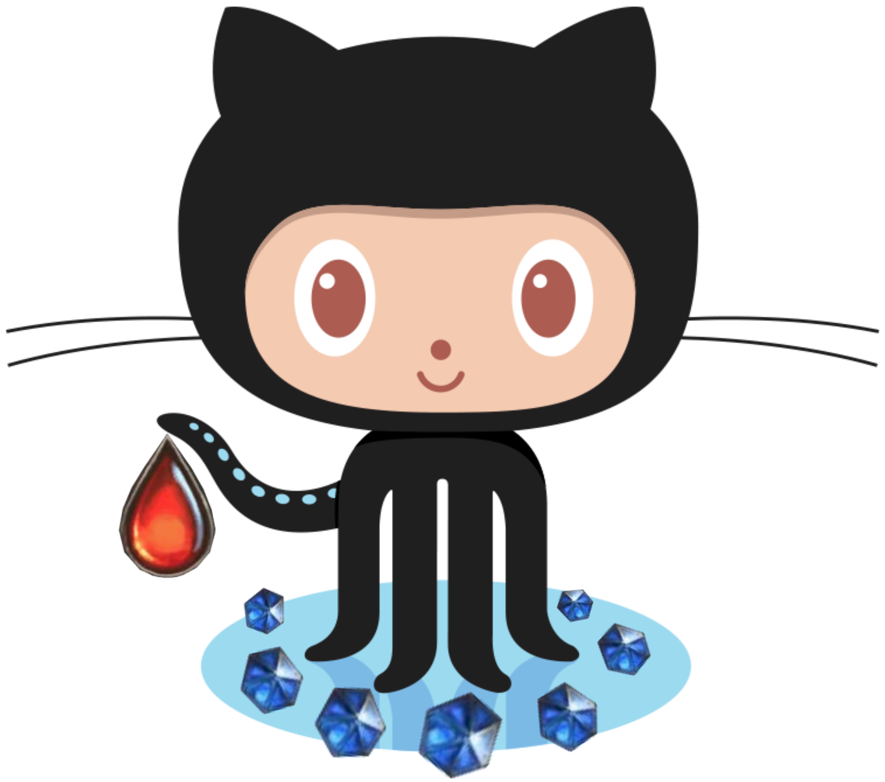


Game End



Kata TCG Rules & Variations

- https://github.com/bkimminich/kata-tcg
  - Detailed rules description
  - Advanced Variations
    - *Healing* cards
    - Use cards as *Minions*
    - Different cost & damage
    - *Card drawer* cards
    - Deck customization

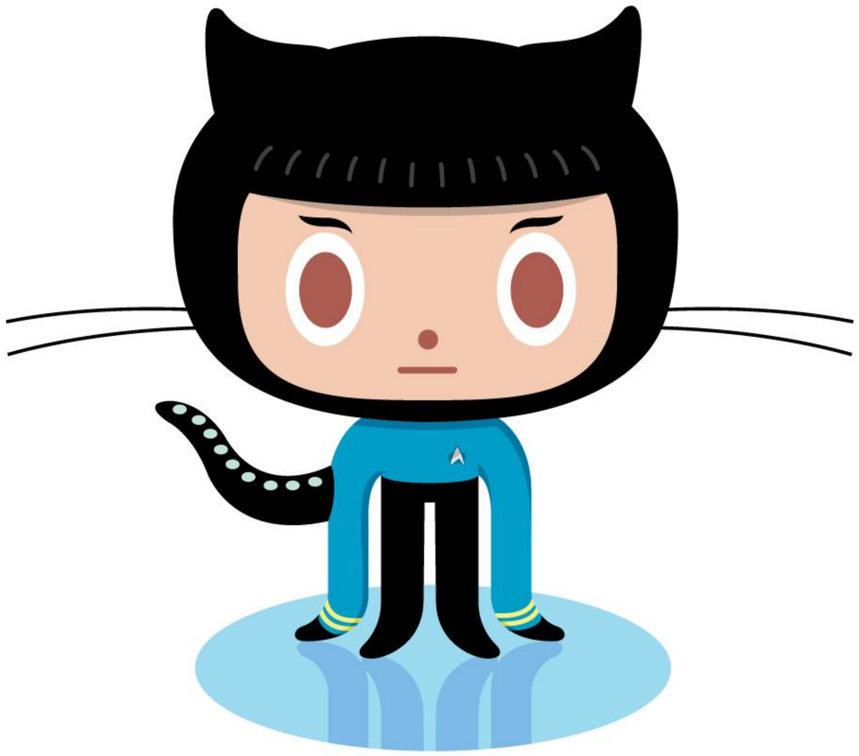


Kata TCG Sample Solution in Java

- https://github.com/bkimminich/kata-tcg
  - Java 8
    - Lambdas & Stream API
  - Junit
  - Mockito
    - For handling dependencies of tested objects
  - Hamcrest
    - Matchers for better legibility in assertions
  - System Rules
    - JUnit @Rules for substituting `java.lang.System`

Other Sample Solutions

- https://github.com/bkimminich/kata-tcg
  - Groovy
    - Spock
  - JavaScript
    - Karma
    - Jasmine
    - PhantomJS

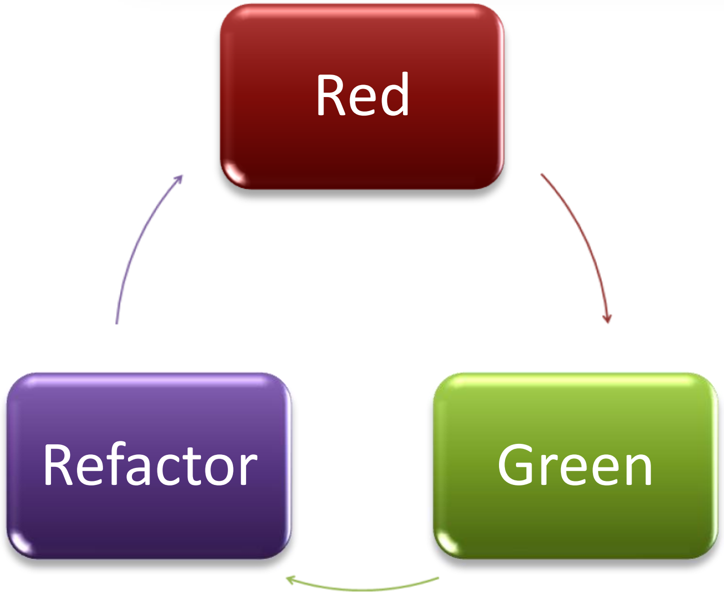


Advanced Unit Testing Examples

Skip No-Brainers

```java
@Test
public void testFrameworkShouldWork() {
    assertThat(true, is(true));
}
```

…



Compile Error as **Red Test**

```java
@Test
public void playerClassShouldExist() {
    Player player = new Player();
}
```

Obvious Implementation

```
@Test
public void playerShouldHave30InitialHealth() {
    assertThat(player.getHealth(), is(equalTo(30)));
}
```

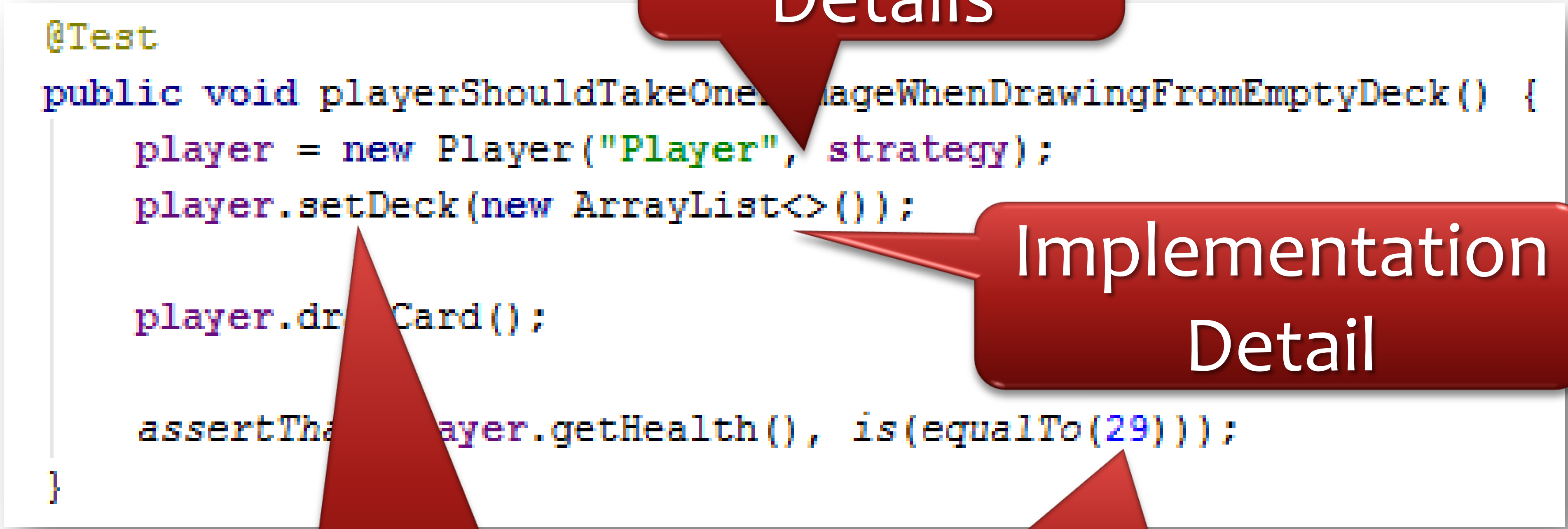
```
public int getHealth() {
    return 30;
}
```

Simplest possible step

```
private int health = 30;

public int getHealth() {
    return health;
}
```

Simplest useful step

Specify the Object under Test



```
@Test
public void playerShouldTakeOne[illegible]ageWhenDrawingFromEmptyDeck() {
    player = new Player("Player", strategy);
    player.setDeck(new ArrayList<>());

    player.dr[illegible]Card();

    assertTha[illegible]ayer.getHealth(), is(equalTo(29)));
}
```

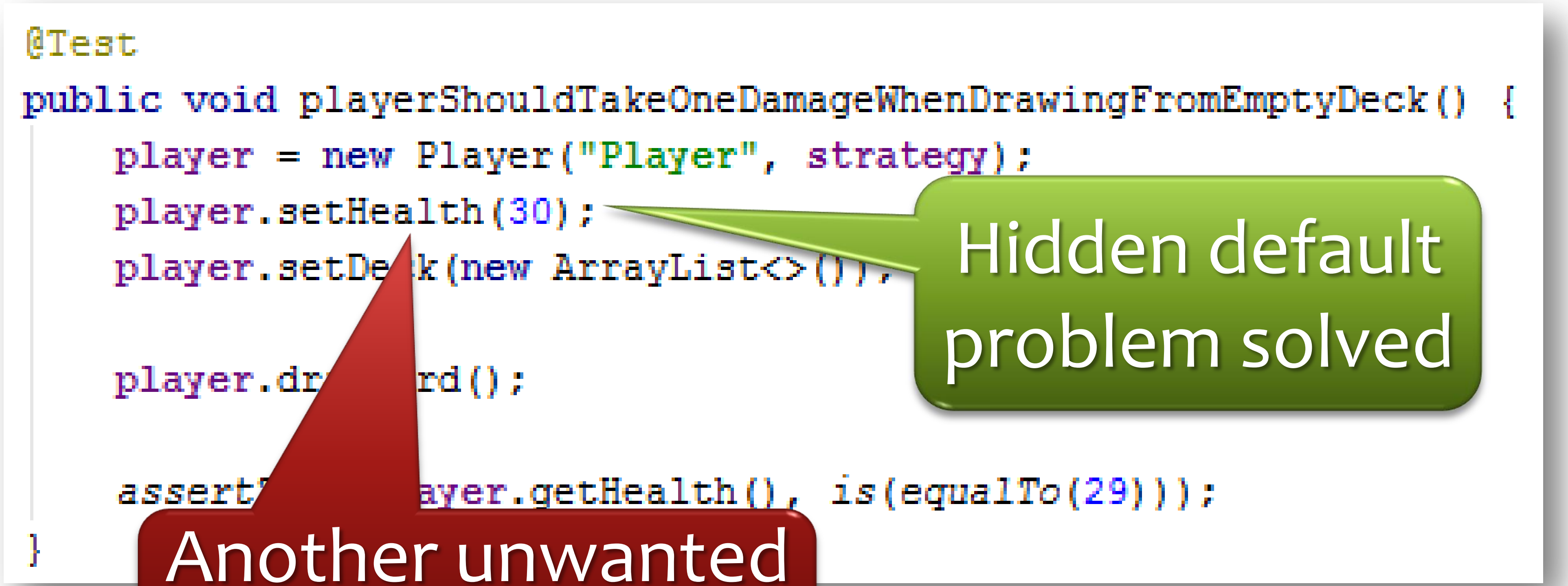
Irrelevant Details

Implementation Detail

Setter introduced for testing

Assuming Hidden Default

Define the Object under Test



```
@Test
public void playerShouldTakeOneDamageWhenDrawingFromEmptyDeck() {
    player = new Player("Player", strategy);
    player.setHealth(30);
    player.setDe k(new ArrayList<>());

    player.dr   rd();

    assert     ayer.getHealth(), is(equalTo(29)));
}
```

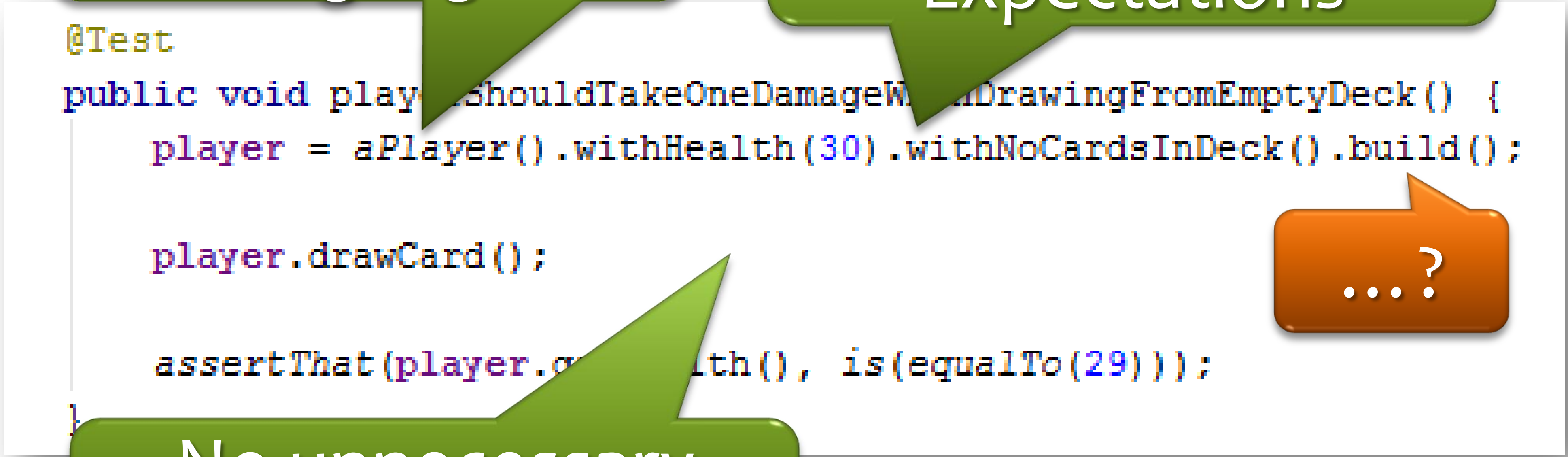
Hidden default problem solved

Another unwanted setter

Builder Pattern

Reads like natural language

Explicitly listed Expectations

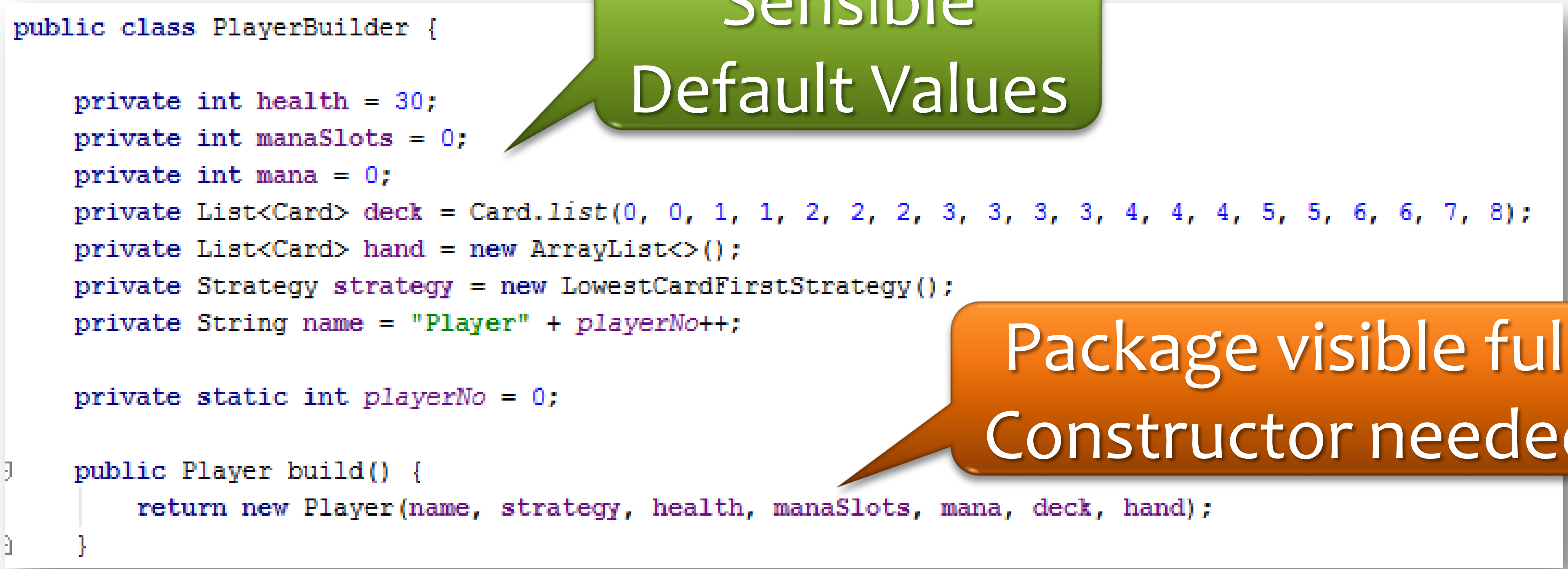


```
@Test
public void play[illegible]ShouldTakeOneDamageW[illegible]nDrawingFromEmptyDeck() {
    player = aPlayer().withHealth(30).withNoCardsInDeck().build();

    player.drawCard();

    assertThat(player.g[illegible]lth(), is(equalTo(29)));
}
```

…?

No unnecessary details

Builder Internals

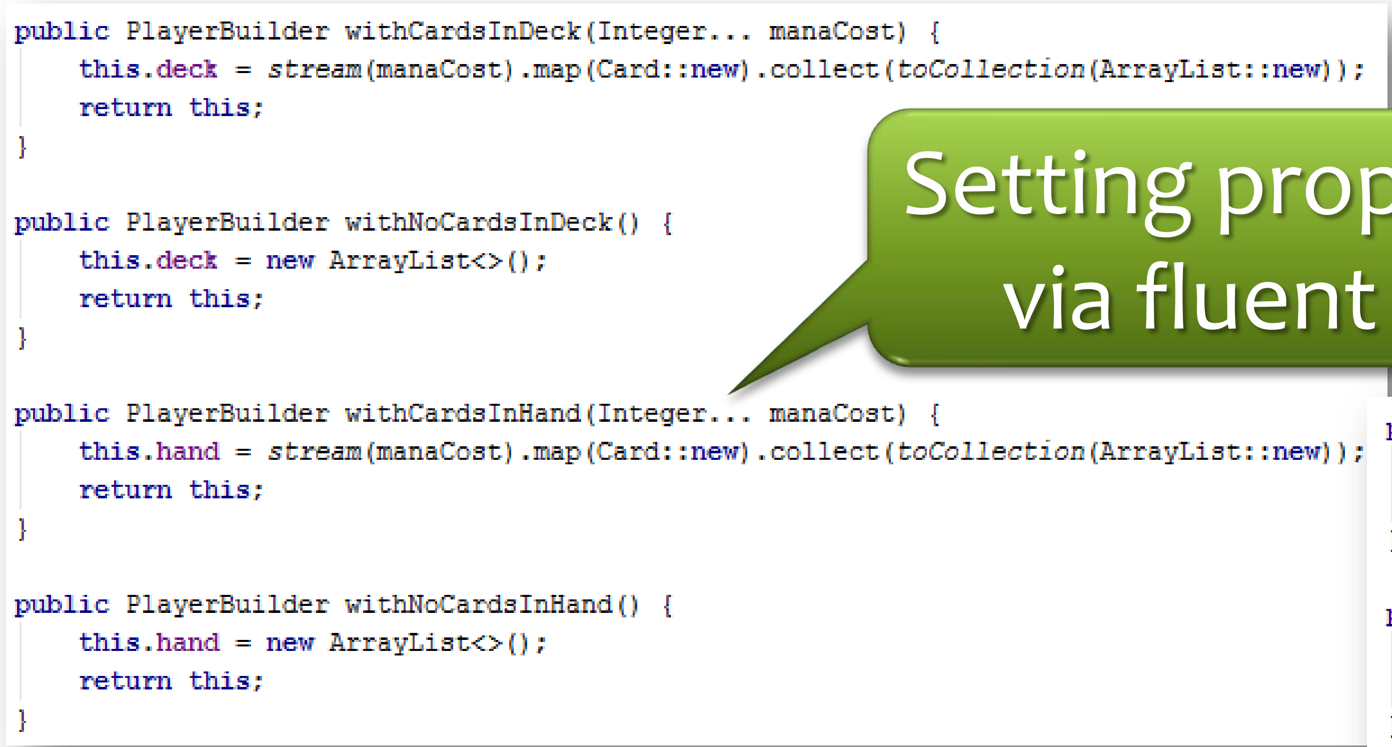


```
public class PlayerBuilder {

    private int health = 30;
    private int manaSlots = 0;
    private int mana = 0;
    private List<Card> deck = Card.list(0, 0, 1, 1, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 5, 5, 6, 6, 7, 8);
    private List<Card> hand = new ArrayList<>();
    private Strategy strategy = new LowestCardFirstStrategy();
    private String name = "Player" + playerNo++;

    private static int playerNo = 0;

    public Player build() {
        return new Player(name, strategy, health, manaSlots, mana, deck, hand);
    }
```

Sensible Default Values

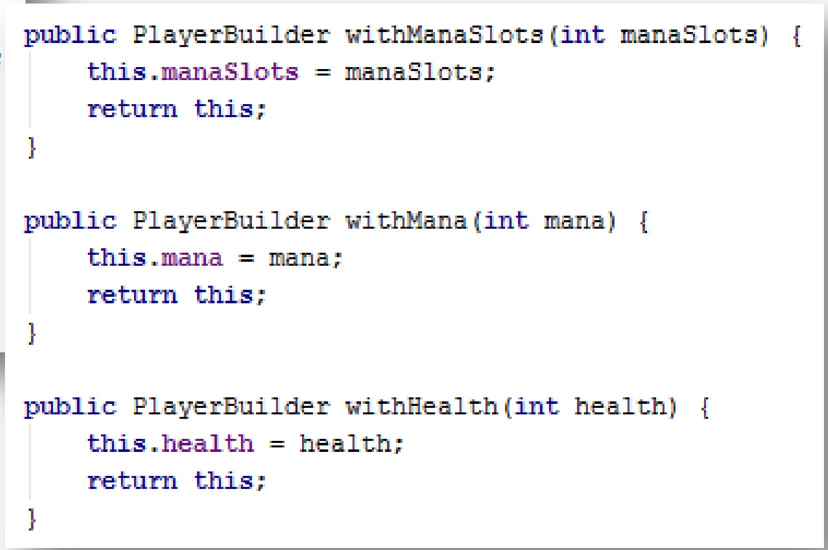
Package visible full Constructor needed

Fluent API



```java
public PlayerBuilder withCardsInDeck(Integer... manaCost) {
    this.deck = stream(manaCost).map(Card::new).collect(toCollection(ArrayList::new));
    return this;
}

public PlayerBuilder withNoCardsInDeck() {
    this.deck = new ArrayList<>();
    return this;
}

public PlayerBuilder withCardsInHand(Integer... manaCost) {
    this.hand = stream(manaCost).map(Card::new).collect(toCollection(ArrayList::new));
    return this;
}

public PlayerBuilder withNoCardsInHand() {
    this.hand = new ArrayList<>();
    return this;
}
```

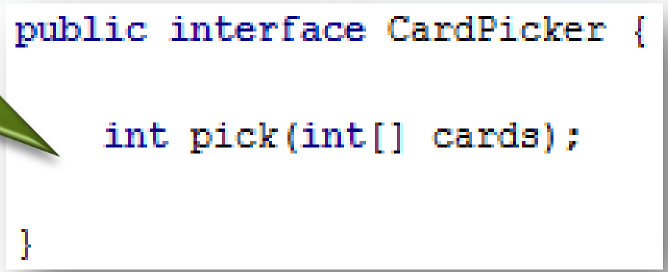
Setting properties via fluent API



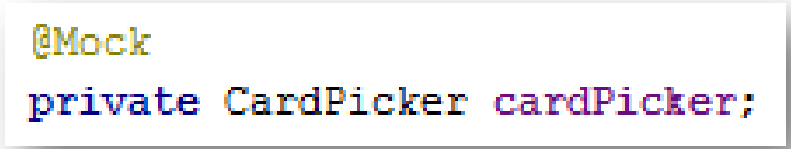
```java
public PlayerBuilder withManaSlots(int manaSlots) {
    this.manaSlots = manaSlots;
    return this;
}

public PlayerBuilder withMana(int mana) {
    this.mana = mana;
    return this;
}

public PlayerBuilder withHealth(int health) {
    this.health = health;
    return this;
}
```

Mocking Behavior

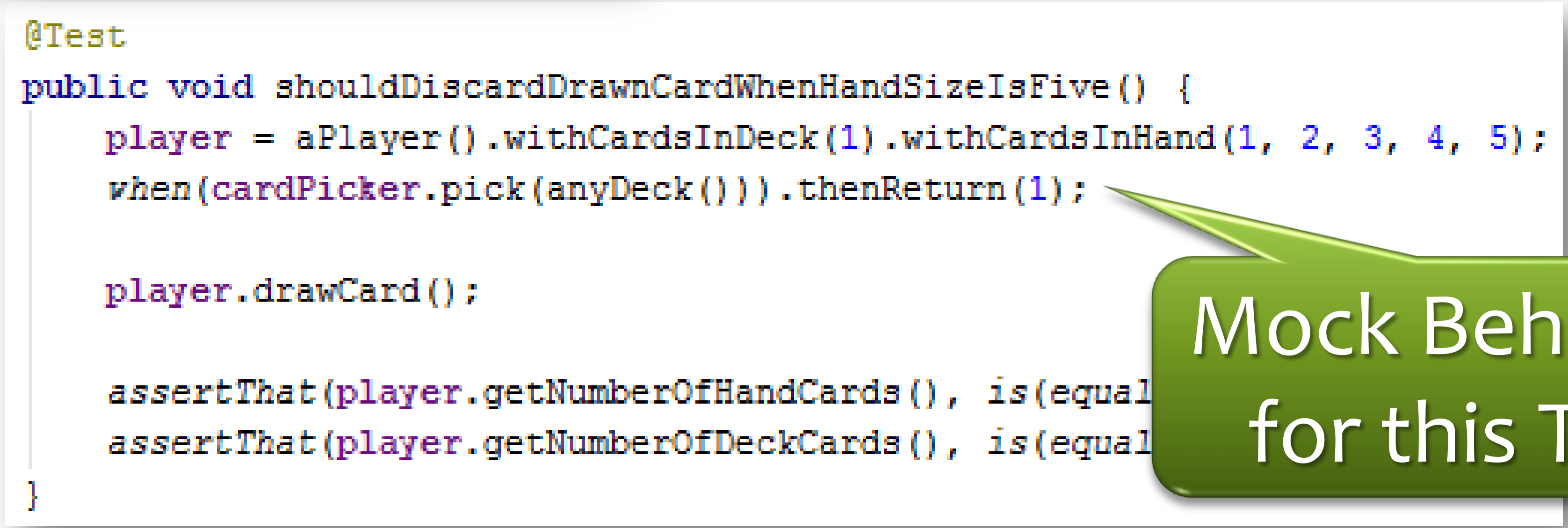
Interface has no implementation yet



```java
public interface CardPicker {

    int pick(int[] cards);

}
```



```java
@Mock
private CardPicker cardPicker;
```

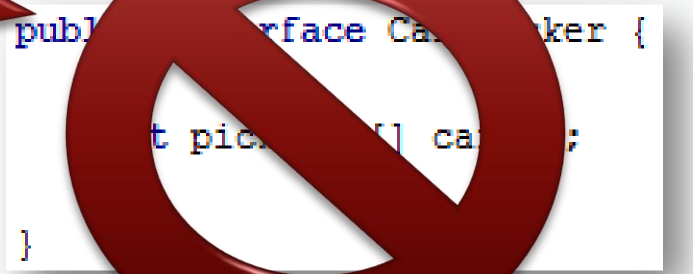


```java
@Test
public void shouldDiscardDrawnCardWhenHandSizeIsFive() {
    player = aPlayer().withCardsInDeck(1).withCardsInHand(1, 2, 3, 4, 5);
    when(cardPicker.pick(anyDeck())).thenReturn(1);

    player.drawCard();

    assertThat(player.getNumberOfHandCards(), is(equal
    assertThat(player.getNumberOfDeckCards(), is(equal
}
```

Mock Behavior for this Test

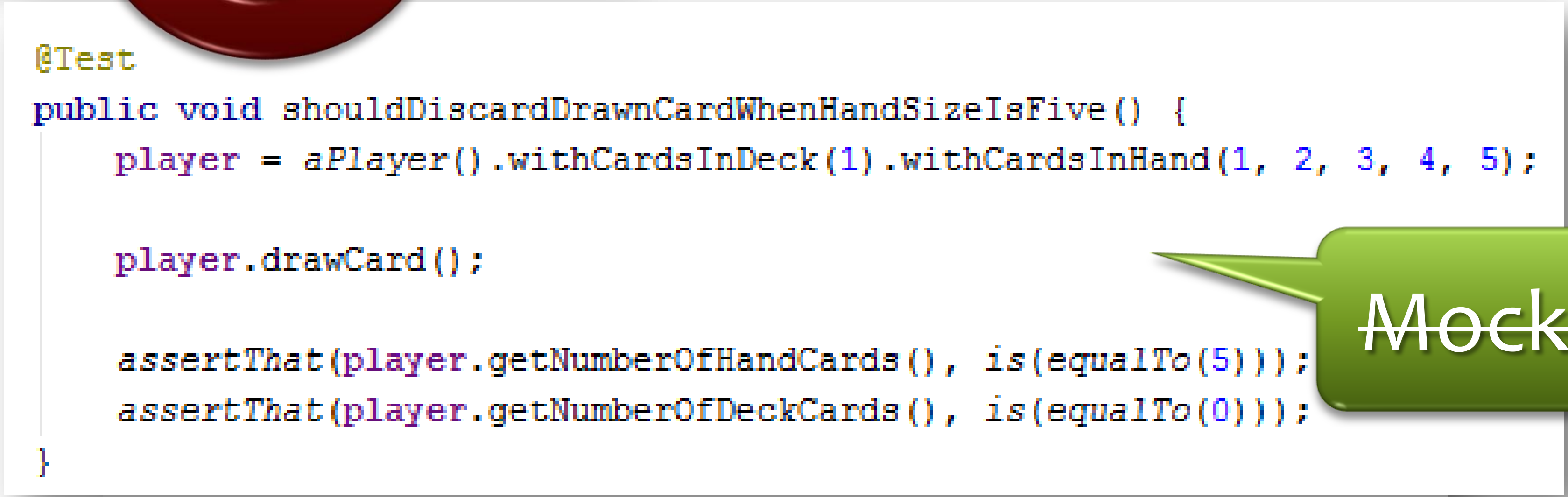
Trashcan Refactoring

CardPicker turned out to be overengineered



```
publ      rface Ca      ker {

     t pic      [] ca      ;

}
```

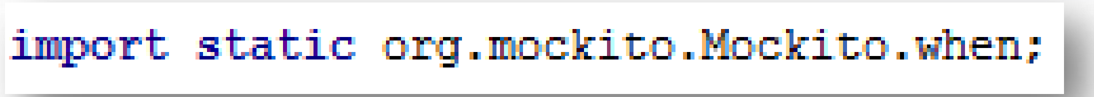
```
@Mock
private    rdPic      ca   Picker;
```



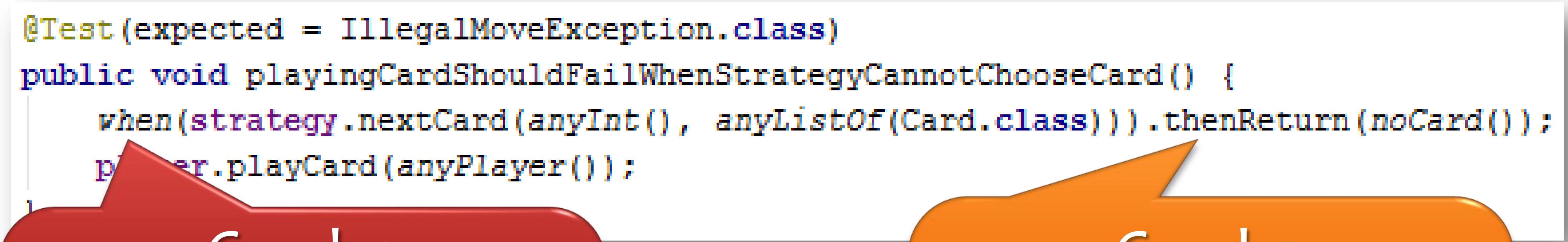
```
@Test
public void shouldDiscardDrawnCardWhenHandSizeIsFive() {
    player = aPlayer().withCardsInDeck(1).withCardsInHand(1, 2, 3, 4, 5);

    player.drawCard();

    assertThat(player.getNumberOfHandCards(), is(equalTo(5)));
    assertThat(player.getNumberOfDeckCards(), is(equalTo(0)));
}
```

~~Mocking~~

Mocking BDD Style



```
import static org.mockito.Mockito.when;
```

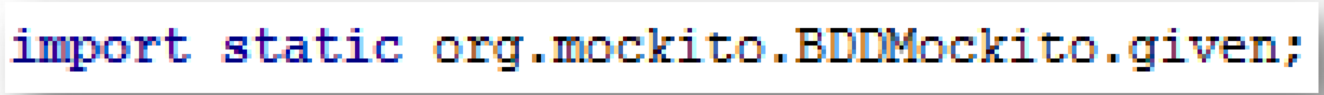


```
@Test(expected = IllegalMoveException.class)
public void playingCardShouldFailWhenStrategyCannotChooseCard() {
    when(strategy.nextCard(anyInt(), anyListOf(Card.class))).thenReturn(noCard());
    pl    er.playCard(anyPlayer());
}
```

Can be confused with Given-**When**-Then part

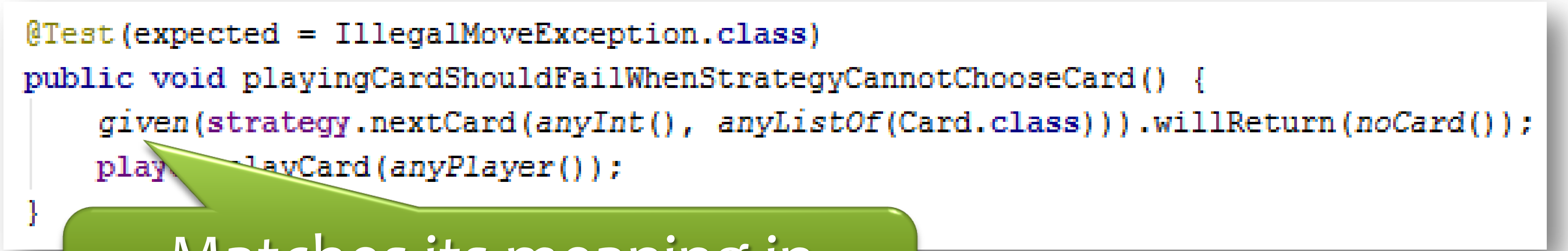
Can be confused with Given-When-**Then** part

Mocking BDD Style



```java
import static org.mockito.BDDMockito.given;
```

Sugar coating for BDD syntax

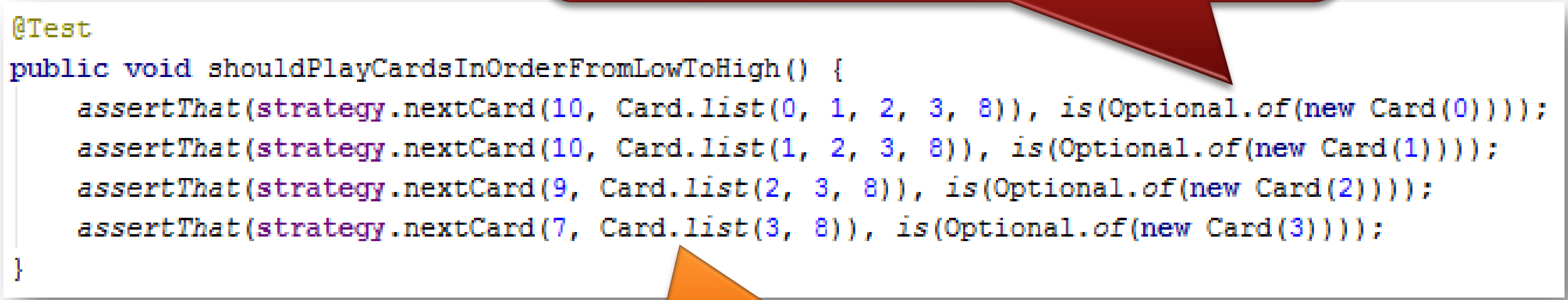


```java
@Test(expected = IllegalMoveException.class)
public void playingCardShouldFailWhenStrategyCannotChooseCard() {
    given(strategy.nextCard(anyInt(), anyListOf(Card.class))).willReturn(noCard());
    play[illegible]layCard(anyPlayer());
}
```

Matches its meaning in **Given**-When-Then structure

Avoid Redundancy

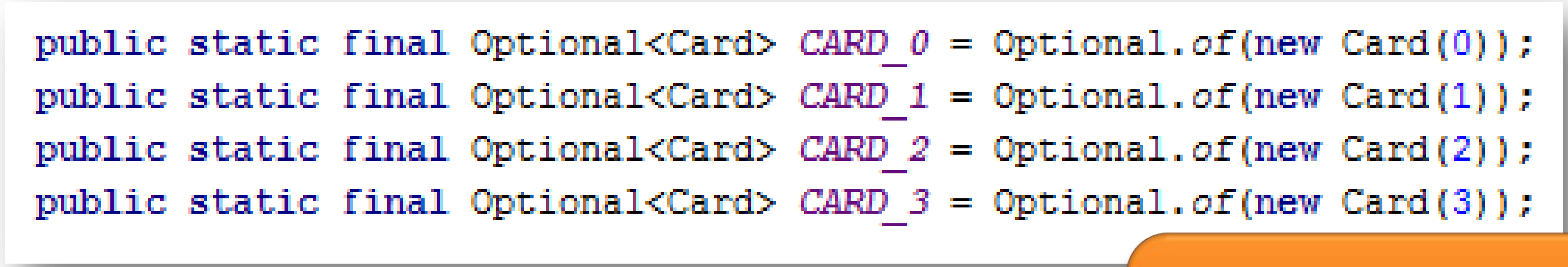
Redundant & spoiled with Implementation Details



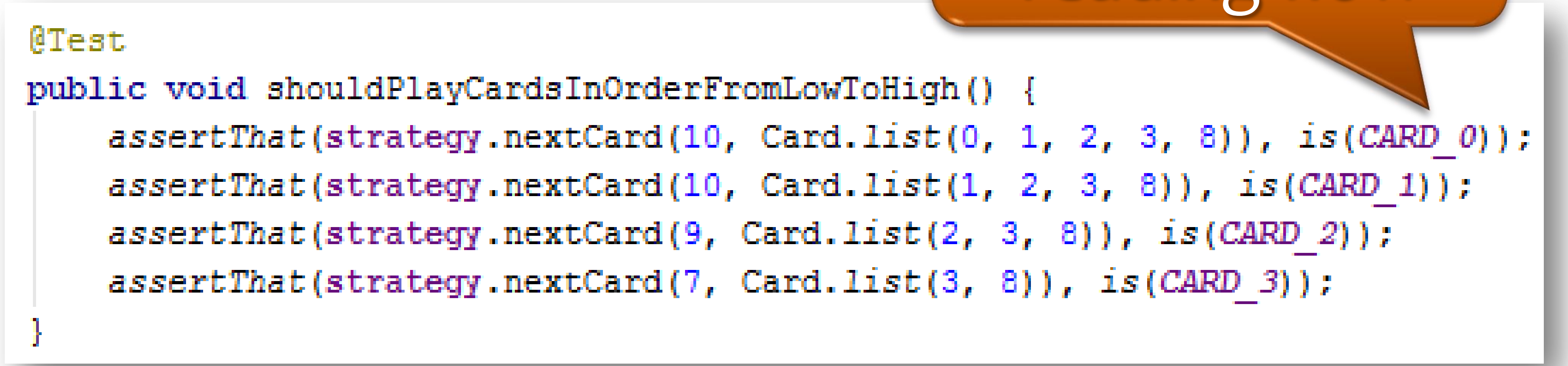
```java
@Test
public void shouldPlayCardsInOrderFromLowToHigh() {
    assertThat(strategy.nextCard(10, Card.list(0, 1, 2, 3, 8)), is(Optional.of(new Card(0))));
    assertThat(strategy.nextCard(10, Card.list(1, 2, 3, 8)), is(Optional.of(new Card(1))));
    assertThat(strategy.nextCard(9, Card.list(2, 3, 8)), is(Optional.of(new Card(2))));
    assertThat(strategy.nextCard(7, Card.list(3, 8)), is(Optional.of(new Card(3))));
}
```

Another slight redundancy

Avoid Constants



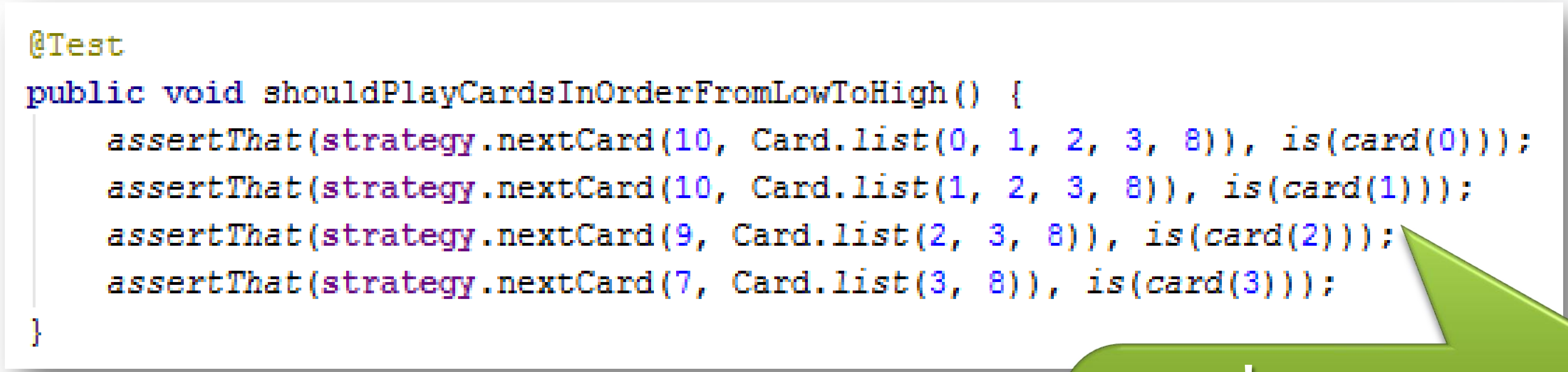
```
public static final Optional<Card> CARD_0 = Optional.of(new Card(0));
public static final Optional<Card> CARD_1 = Optional.of(new Card(1));
public static final Optional<Card> CARD_2 = Optional.of(new Card(2));
public static final Optional<Card> CARD_3 = Optional.of(new Card(3));
```



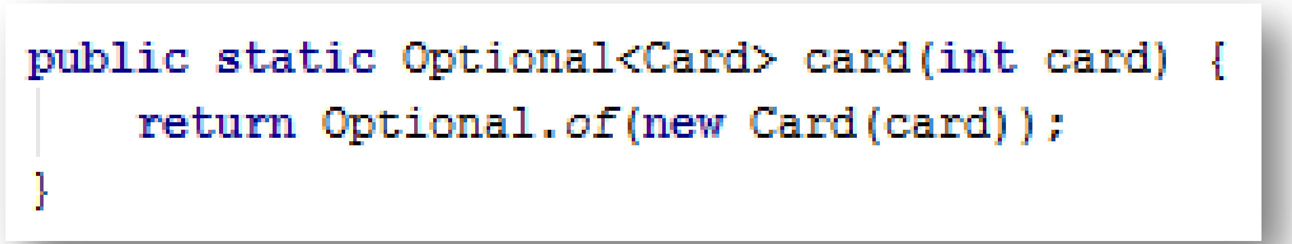
```
@Test
public void shouldPlayCardsInOrderFromLowToHigh() {
    assertThat(strategy.nextCard(10, Card.list(0, 1, 2, 3, 8)), is(CARD_0));
    assertThat(strategy.nextCard(10, Card.list(1, 2, 3, 8)), is(CARD_1));
    assertThat(strategy.nextCard(9, Card.list(2, 3, 8)), is(CARD_2));
    assertThat(strategy.nextCard(7, Card.list(3, 8)), is(CARD_3));
}
```

Disturbs the reading flow

Syntactic Sugar



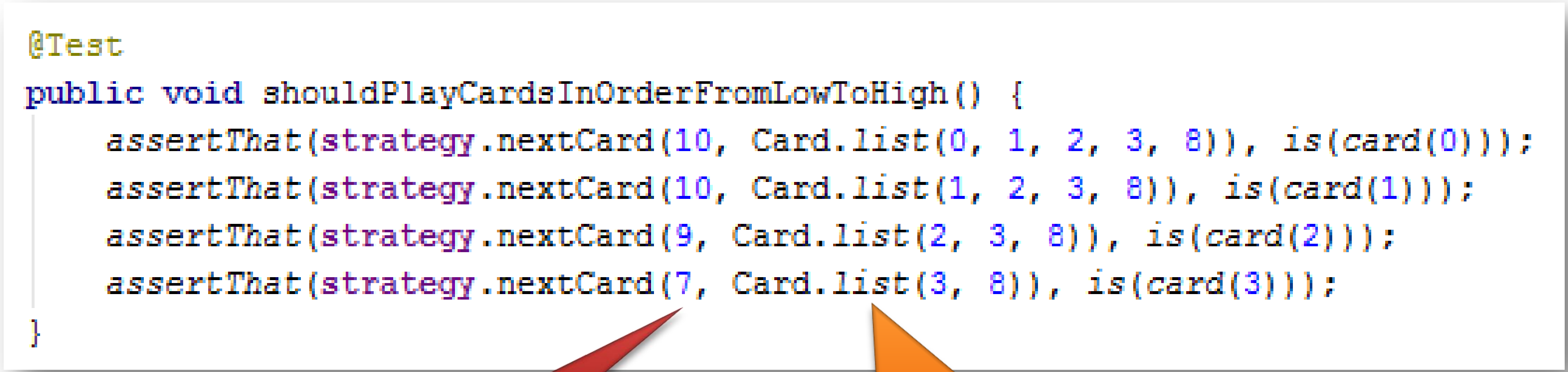
```java
@Test
public void shouldPlayCardsInOrderFromLowToHigh() {
    assertThat(strategy.nextCard(10, Card.list(0, 1, 2, 3, 8)), is(card(0)));
    assertThat(strategy.nextCard(10, Card.list(1, 2, 3, 8)), is(card(1)));
    assertThat(strategy.nextCard(9, Card.list(2, 3, 8)), is(card(2)));
    assertThat(strategy.nextCard(7, Card.list(3, 8)), is(card(3)));
}
```



```java
public static Optional<Card> card(int card) {
    return Optional.of(new Card(card));
}
```

Improves Legibility and removes Redundancy

More Syntactic Sugar

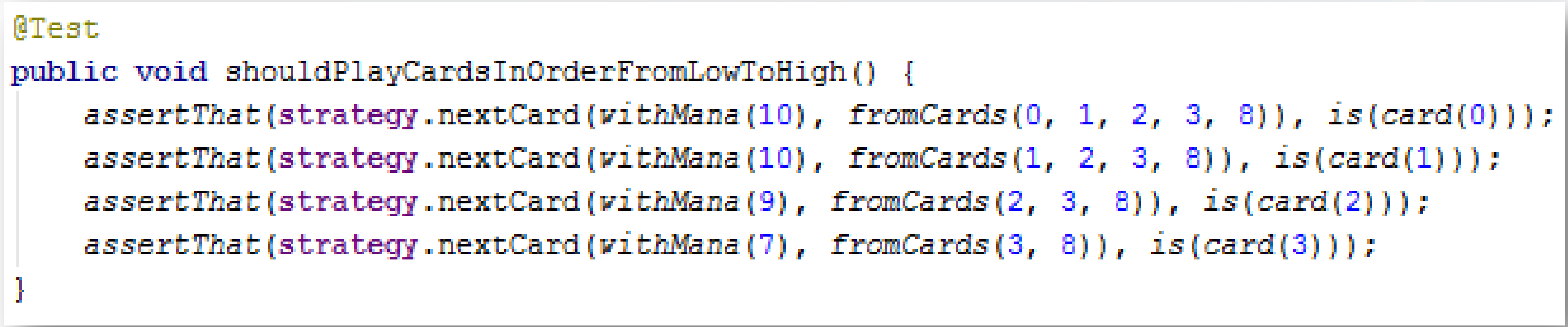


```java
@Test
public void shouldPlayCardsInOrderFromLowToHigh() {
    assertThat(strategy.nextCard(10, Card.list(0, 1, 2, 3, 8)), is(card(0)));
    assertThat(strategy.nextCard(10, Card.list(1, 2, 3, 8)), is(card(1)));
    assertThat(strategy.nextCard(9, Card.list(2, 3, 8)), is(card(2)));
    assertThat(strategy.nextCard(7, Card.list(3, 8)), is(card(3)));
}
```

Magic Numbers

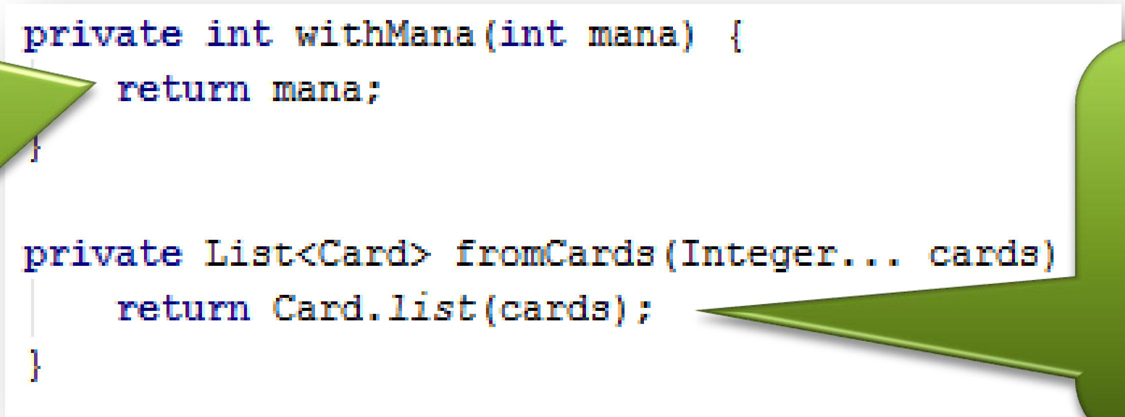
Different Level of Abstraction

Syntactic Artificial Sweetener



```java
@Test
public void shouldPlayCardsInOrderFromLowToHigh() {
    assertThat(strategy.nextCard(withMana(10), fromCards(0, 1, 2, 3, 8)), is(card(0)));
    assertThat(strategy.nextCard(withMana(10), fromCards(1, 2, 3, 8)), is(card(1)));
    assertThat(strategy.nextCard(withMana(9), fromCards(2, 3, 8)), is(card(2)));
    assertThat(strategy.nextCard(withMana(7), fromCards(3, 8)), is(card(3)));
}
```

No-Op Syntactic Sugar for Legibility

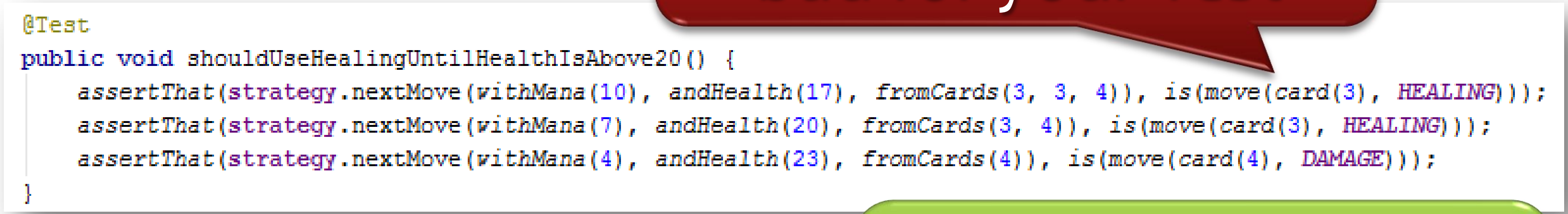


```java
private int withMana(int mana) {
    return mana;
}

private List<Card> fromCards(Integer... cards)
    return Card.list(cards);
}
```

Syntactic Sugar reduces Redundancy

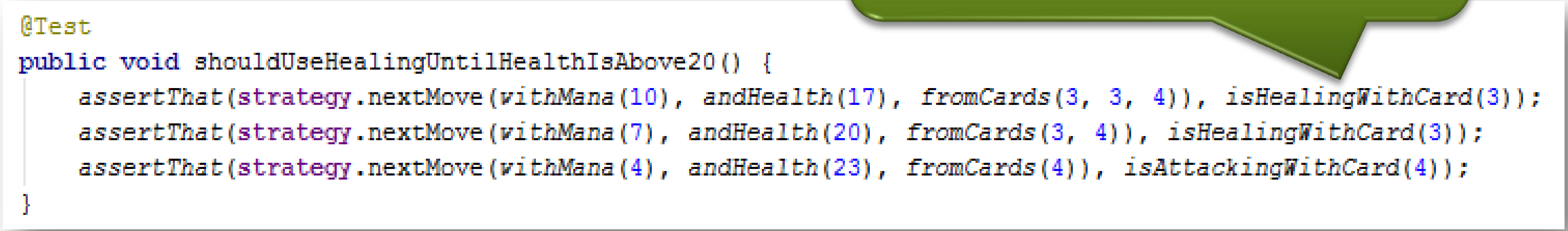
Test Diabetes

Too much Sugar is bad for your Test



```
@Test
public void shouldUseHealingUntilHealthIsAbove20() {
    assertThat(strategy.nextMove(withMana(10), andHealth(17), fromCards(3, 3, 4)), is(move(card(3), HEALING)));
    assertThat(strategy.nextMove(withMana(7), andHealth(20), fromCards(3, 4)), is(move(card(3), HEALING)));
    assertThat(strategy.nextMove(withMana(4), andHealth(23), fromCards(4)), is(move(card(4), DAMAGE)));
}
```

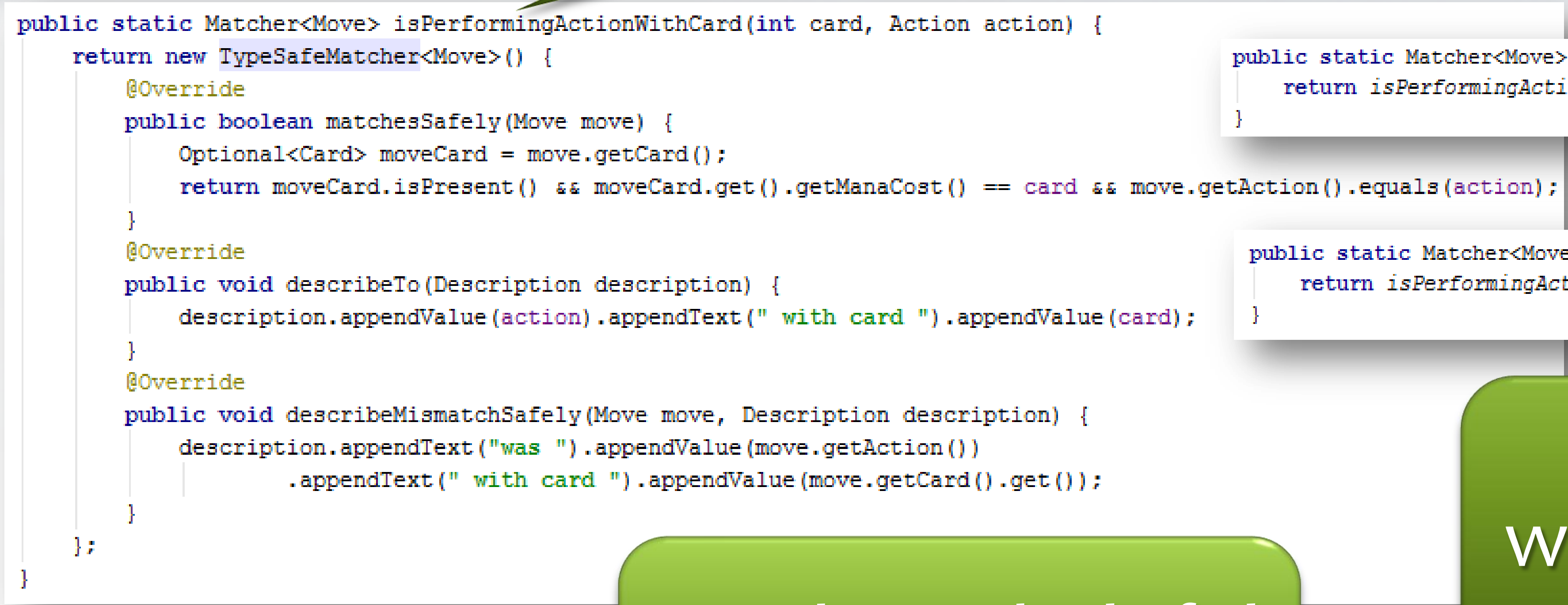
Customer Matchers can medicate this



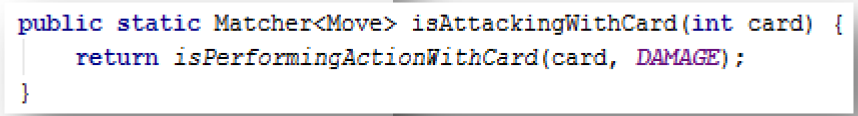
```
@Test
public void shouldUseHealingUntilHealthIsAbove20() {
    assertThat(strategy.nextMove(withMana(10), andHealth(17), fromCards(3, 3, 4)), isHealingWithCard(3));
    assertThat(strategy.nextMove(withMana(7), andHealth(20), fromCards(3, 4)), isHealingWithCard(3));
    assertThat(strategy.nextMove(withMana(4), andHealth(23), fromCards(4)), isAttackingWithCard(4));
}
```

Custom Matcher

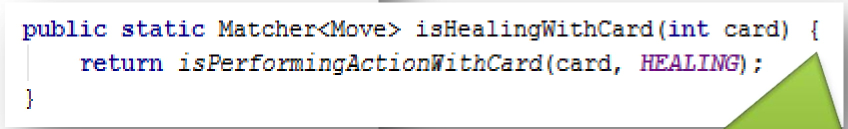
Encapsulation of Expectation



```java
public static Matcher<Move> isPerformingActionWithCard(int card, Action action) {
    return new TypeSafeMatcher<Move>() {
        @Override
        public boolean matchesSafely(Move move) {
            Optional<Card> moveCard = move.getCard();
            return moveCard.isPresent() && moveCard.get().getManaCost() == card && move.getAction().equals(action);
        }
        @Override
        public void describeTo(Description description) {
            description.appendValue(action).appendText(" with card ").appendValue(card);
        }
        @Override
        public void describeMismatchSafely(Move move, Description description) {
            description.appendText("was ").appendValue(move.getAction())
                    .appendText(" with card ").appendValue(move.getCard().get());
        }
    };
}
```



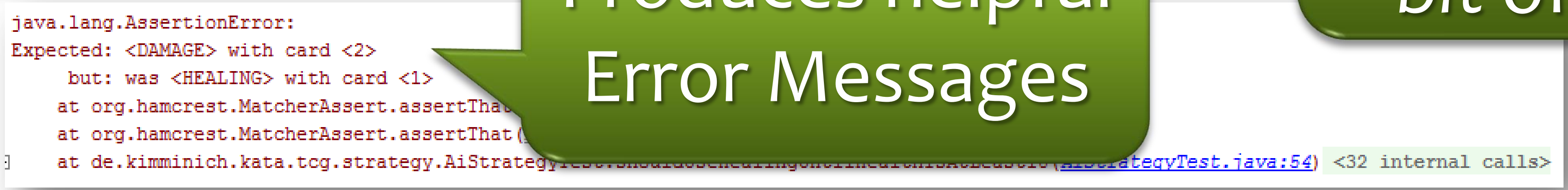
```java
public static Matcher<Move> isAttackingWithCard(int card) {
    return isPerformingActionWithCard(card, DAMAGE);
}
```



```java
public static Matcher<Move> isHealingWithCard(int card) {
    return isPerformingActionWithCard(card, HEALING);
}
```

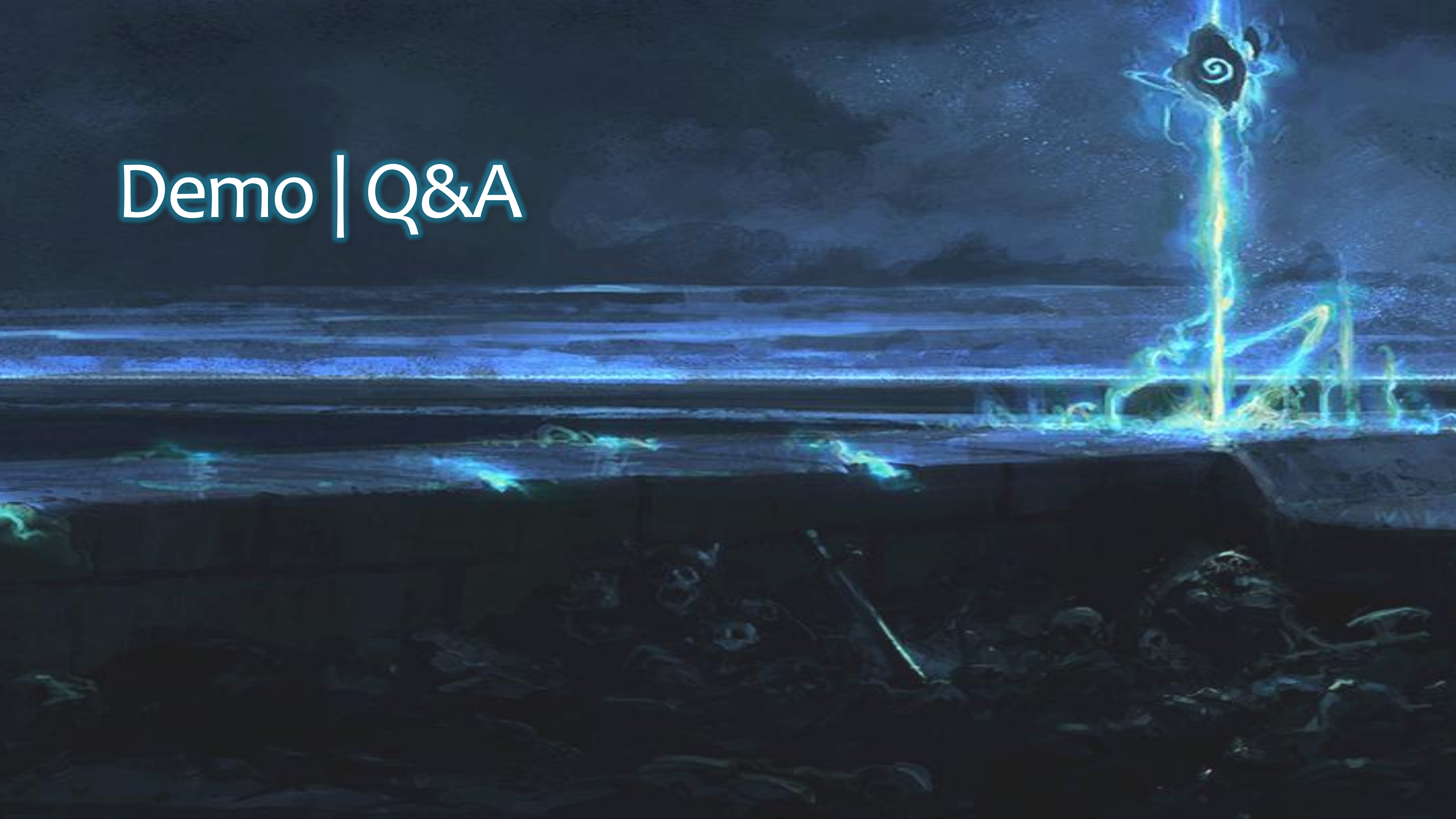
Even better with *just a little bit* of Sugar

Produces helpful Error Messages

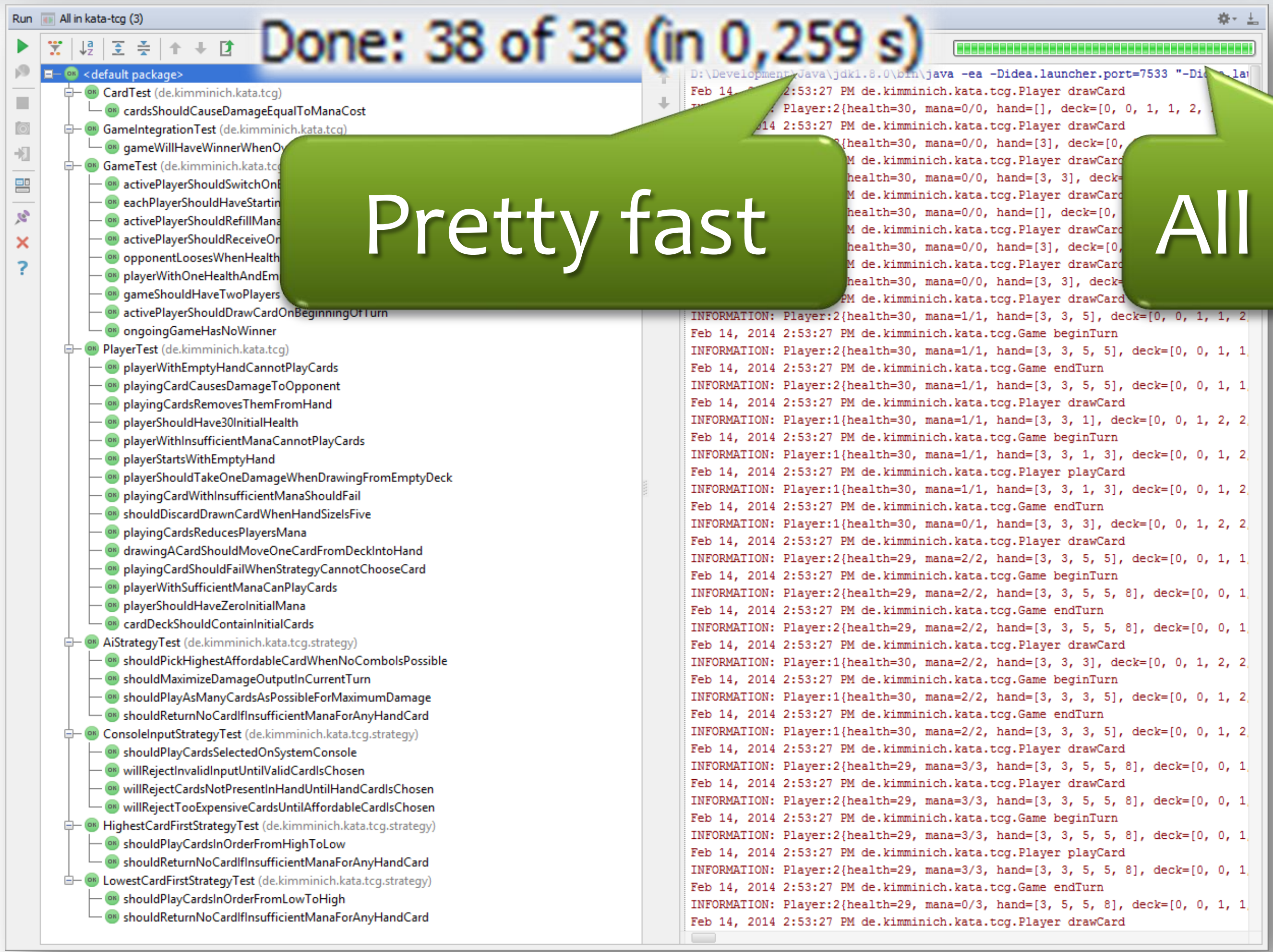


```
java.lang.AssertionError:
Expected: <DAMAGE> with card <2>
     but: was <HEALING> with card <1>
    at org.hamcrest.MatcherAssert.assertTha
    at org.hamcrest.MatcherAssert.assertThat(
]   at de.kimminich.kata.tcg.strategy.AiStrategy[illegible](AiStrategyTest.java:54) <32 internal calls>
```

Demo | Q&A



Demo: Test Execution



Run   All in kata-tcg (3)

Done: 38 of 38 (in 0,259 s)

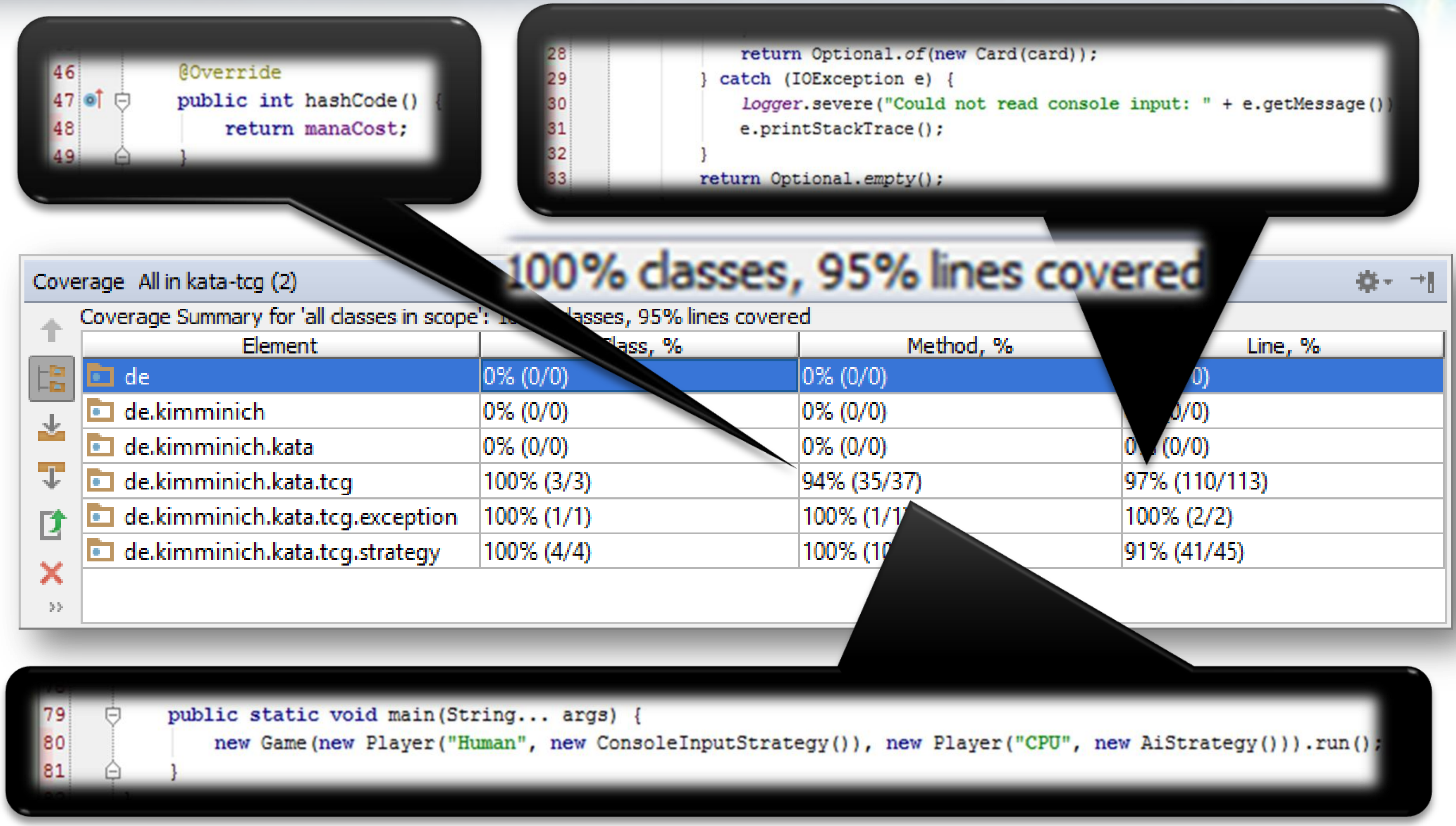
```
<default package>
  CardTest (de.kimminich.kata.tcg)
    cardsShouldCauseDamageEqualToManaCost
  GameIntegrationTest (de.kimminich.kata.tcg)
    gameWillHaveWinnerWhenO
  GameTest (de.kimminich.kata.tcg
    activePlayerShouldSwitchOn
    eachPlayerShouldHaveStartin
    activePlayerShouldRefillMana
    activePlayerShouldReceiveO
    opponentLoosesWhenHealth
    playerWithOneHealthAndEm
    gameShouldHaveTwoPlayers
    activePlayerShouldDrawCardOnBeginningOfTurn
    ongoingGameHasNoWinner
  PlayerTest (de.kimminich.kata.tcg)
    playerWithEmptyHandCannotPlayCards
    playingCardCausesDamageToOpponent
    playingCardsRemovesThemFromHand
    playerShouldHave30InitialHealth
    playerWithInsufficientManaCannotPlayCards
    playerStartsWithEmptyHand
    playerShouldTakeOneDamageWhenDrawingFromEmptyDeck
    playingCardWithInsufficientManaShouldFail
    shouldDiscardDrawnCardWhenHandSizeIsFive
    playingCardsReducesPlayersMana
    drawingACardShouldMoveOneCardFromDeckIntoHand
    playingCardShouldFailWhenStrategyCannotChooseCard
    playerWithSufficientManaCanPlayCards
    playerShouldHaveZeroInitialMana
    cardDeckShouldContainInitialCards
  AiStrategyTest (de.kimminich.kata.tcg.strategy)
    shouldPickHighestAffordableCardWhenNoComboIsPossible
    shouldMaximizeDamageOutputInCurrentTurn
    shouldPlayAsManyCardsAsPossibleForMaximumDamage
    shouldReturnNoCardIfInsufficientManaForAnyHandCard
  ConsoleInputStrategyTest (de.kimminich.kata.tcg.strategy)
    shouldPlayCardsSelectedOnSystemConsole
    willRejectInvalidInputUntilValidCardIsChosen
    willRejectCardsNotPresentInHandUntilHandCardIsChosen
    willRejectTooExpensiveCardsUntilAffordableCardIsChosen
  HighestCardFirstStrategyTest (de.kimminich.kata.tcg.strategy)
    shouldPlayCardsInOrderFromHighToLow
    shouldReturnNoCardIfInsufficientManaForAnyHandCard
  LowestCardFirstStrategyTest (de.kimminich.kata.tcg.strategy)
    shouldPlayCardsInOrderFromLowToHigh
    shouldReturnNoCardIfInsufficientManaForAnyHandCard
```

```
D:\Development\Java\jdk1.8.0\bin\java -ea -Didea.launcher.port=7533 "-Di
Feb 14,       2:53:27 PM de.kimminich.kata.tcg.Player drawCard
          : Player:2{health=30, mana=0/0, hand=[], deck=[0, 0, 1, 1, 2,
         014 2:53:27 PM de.kimminich.kata.tcg.Player drawCard
              {health=30, mana=0/0, hand=[3], deck=[0,
                M de.kimminich.kata.tcg.Player drawCard
               health=30, mana=0/0, hand=[3, 3], deck
                M de.kimminich.kata.tcg.Player drawCard
               health=30, mana=0/0, hand=[], deck=[0,
                M de.kimminich.kata.tcg.Player drawCard
               health=30, mana=0/0, hand=[3], deck=[0,
                M de.kimminich.kata.tcg.Player drawCard
               health=30, mana=0/0, hand=[3, 3], deck
                PM de.kimminich.kata.tcg.Player drawCard
INFORMATION: Player:2{health=30, mana=1/1, hand=[3, 3, 5], deck=[0, 0, 1, 1, 2
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Game beginTurn
INFORMATION: Player:2{health=30, mana=1/1, hand=[3, 3, 5, 5], deck=[0, 0, 1, 1
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Game endTurn
INFORMATION: Player:2{health=30, mana=1/1, hand=[3, 3, 5, 5], deck=[0, 0, 1, 1
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Player drawCard
INFORMATION: Player:1{health=30, mana=1/1, hand=[3, 3, 1], deck=[0, 0, 1, 2, 2
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Game beginTurn
INFORMATION: Player:1{health=30, mana=1/1, hand=[3, 3, 1, 3], deck=[0, 0, 1, 2
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Player playCard
INFORMATION: Player:1{health=30, mana=1/1, hand=[3, 3, 1, 3], deck=[0, 0, 1, 2
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Game endTurn
INFORMATION: Player:1{health=30, mana=0/1, hand=[3, 3, 3], deck=[0, 0, 1, 2, 2
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Player drawCard
INFORMATION: Player:2{health=29, mana=2/2, hand=[3, 3, 5, 5], deck=[0, 0, 1, 1
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Game beginTurn
INFORMATION: Player:2{health=29, mana=2/2, hand=[3, 3, 5, 5, 8], deck=[0, 0, 1
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Game endTurn
INFORMATION: Player:2{health=29, mana=2/2, hand=[3, 3, 5, 5, 8], deck=[0, 0, 1
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Player drawCard
INFORMATION: Player:1{health=30, mana=2/2, hand=[3, 3, 3], deck=[0, 0, 1, 2, 2
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Game beginTurn
INFORMATION: Player:1{health=30, mana=2/2, hand=[3, 3, 3, 5], deck=[0, 0, 1, 2
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Game endTurn
INFORMATION: Player:1{health=30, mana=2/2, hand=[3, 3, 3, 5], deck=[0, 0, 1, 2
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Player drawCard
INFORMATION: Player:2{health=29, mana=3/3, hand=[3, 3, 5, 5, 8], deck=[0, 0, 1
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Player drawCard
INFORMATION: Player:2{health=29, mana=3/3, hand=[3, 3, 5, 5, 8], deck=[0, 0, 1
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Game beginTurn
INFORMATION: Player:2{health=29, mana=3/3, hand=[3, 3, 5, 5, 8], deck=[0, 0, 1
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Player playCard
INFORMATION: Player:2{health=29, mana=3/3, hand=[3, 3, 5, 5, 8], deck=[0, 0, 1
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Game endTurn
INFORMATION: Player:2{health=29, mana=0/3, hand=[3, 5, 5, 8], deck=[0, 0, 1, 1
Feb 14, 2014 2:53:27 PM de.kimminich.kata.tcg.Player drawCard
```

Pretty fast

All tests pass

Demo: Code Coverage



```
    @Override
    public int hashCode() {
        return manaCost;
    }
```

```
            return Optional.of(new Card(card));
        } catch (IOException e) {
            logger.severe("Could not read console input: " + e.getMessage())
            e.printStackTrace();
        }
        return Optional.empty();
```

100% classes, 95% lines covered

Coverage All in kata-tcg (2)

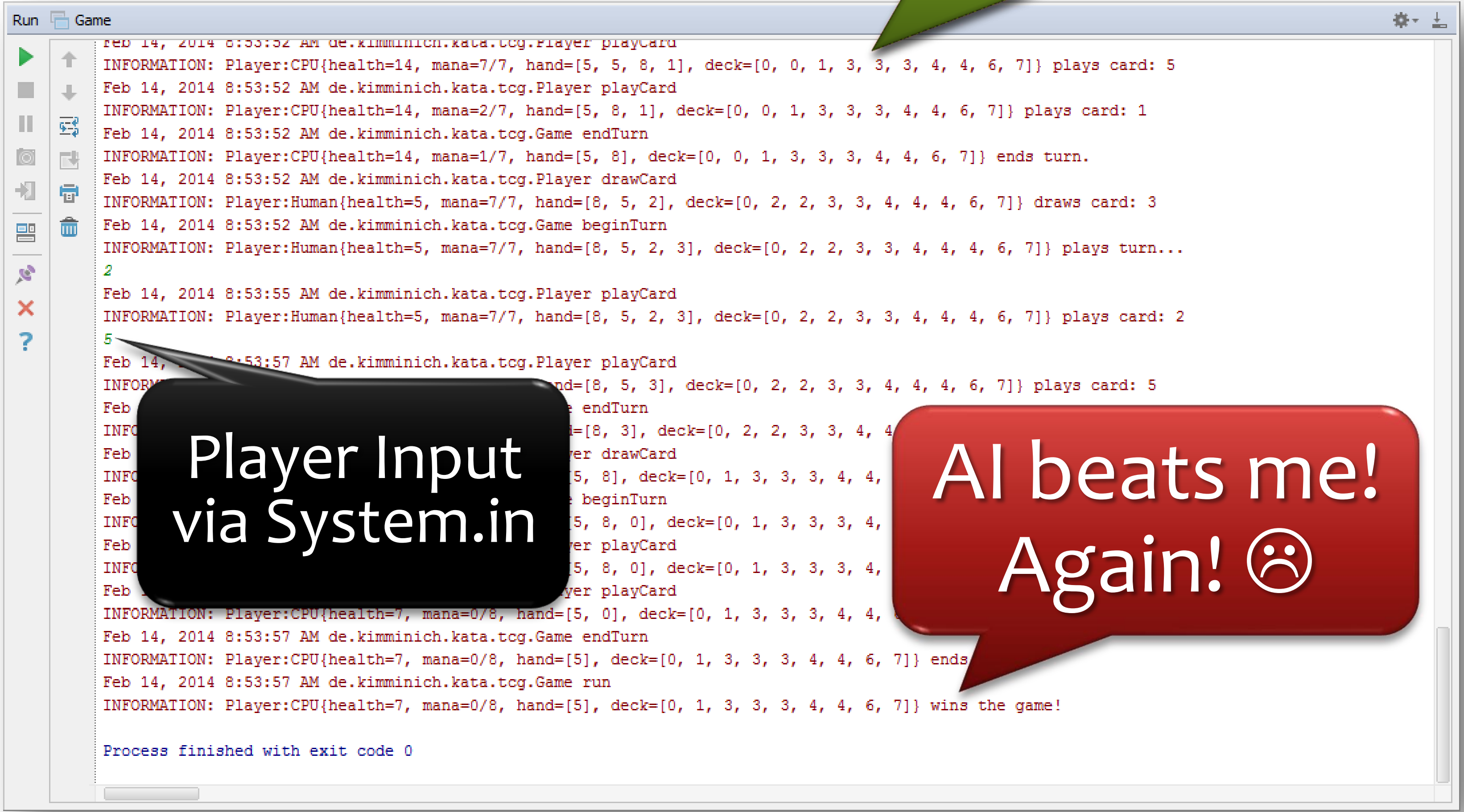
Coverage Summary for 'all classes in scope': 1[illegible]lasses, 95% lines covered

Element	Class, %	Method, %	Line, %
de	0% (0/0)	0% (0/0)	[illegible]0)
de.kimminich	0% (0/0)	0% (0/0)	[illegible]0/0)
de.kimminich.kata	0% (0/0)	0% (0/0)	0[illegible] (0/0)
de.kimminich.kata.tcg	100% (3/3)	94% (35/37)	97% (110/113)
de.kimminich.kata.tcg.exception	100% (1/1)	100% (1/1[illegible]	100% (2/2)
de.kimminich.kata.tcg.strategy	100% (4/4)	100% (1[illegible]	91% (41/45)

```
    public static void main(String... args) {
        new Game(new Player("Human", new ConsoleInputStrategy()), new Player("CPU", new AiStrategy())).run();
    }
```

Demo: Human vs. AI Game

~~Ugly~~ Retro UI



```
Feb 14, 2014 8:53:52 AM de.kimminich.kata.tcg.Player playCard
INFORMATION: Player:CPU{health=14, mana=7/7, hand=[5, 5, 8, 1], deck=[0, 0, 1, 3, 3, 3, 4, 4, 6, 7]} plays card: 5
Feb 14, 2014 8:53:52 AM de.kimminich.kata.tcg.Player playCard
INFORMATION: Player:CPU{health=14, mana=2/7, hand=[5, 8, 1], deck=[0, 0, 1, 3, 3, 3, 4, 4, 6, 7]} plays card: 1
Feb 14, 2014 8:53:52 AM de.kimminich.kata.tcg.Game endTurn
INFORMATION: Player:CPU{health=14, mana=1/7, hand=[5, 8], deck=[0, 0, 1, 3, 3, 3, 4, 4, 6, 7]} ends turn.
Feb 14, 2014 8:53:52 AM de.kimminich.kata.tcg.Player drawCard
INFORMATION: Player:Human{health=5, mana=7/7, hand=[8, 5, 2], deck=[0, 2, 2, 3, 3, 4, 4, 4, 6, 7]} draws card: 3
Feb 14, 2014 8:53:52 AM de.kimminich.kata.tcg.Game beginTurn
INFORMATION: Player:Human{health=5, mana=7/7, hand=[8, 5, 2, 3], deck=[0, 2, 2, 3, 3, 4, 4, 4, 6, 7]} plays turn...
2
Feb 14, 2014 8:53:55 AM de.kimminich.kata.tcg.Player playCard
INFORMATION: Player:Human{health=5, mana=7/7, hand=[8, 5, 2, 3], deck=[0, 2, 2, 3, 3, 4, 4, 4, 6, 7]} plays card: 2
5
Feb 14,      :53:57 AM de.kimminich.kata.tcg.Player playCard
INFOR                          nd=[8, 5, 3], deck=[0, 2, 2, 3, 3, 4, 4, 4, 6, 7]} plays card: 5
Feb                             endTurn
INFO                           =[8, 3], deck=[0, 2, 2, 3, 3, 4, 4
Feb                            er drawCard
INFO                           5, 8], deck=[0, 1, 3, 3, 3, 4, 4,
Feb                             beginTurn
INFO                           5, 8, 0], deck=[0, 1, 3, 3, 3, 4,
Feb                            er playCard
INFO                           5, 8, 0], deck=[0, 1, 3, 3, 3, 4,
Feb                            yer playCard
INFORMATION: Player:CPU{health=7, mana=0/8, hand=[5, 0], deck=[0, 1, 3, 3, 3, 4, 4,
Feb 14, 2014 8:53:57 AM de.kimminich.kata.tcg.Game endTurn
INFORMATION: Player:CPU{health=7, mana=0/8, hand=[5], deck=[0, 1, 3, 3, 3, 4, 4, 6, 7]} ends
Feb 14, 2014 8:53:57 AM de.kimminich.kata.tcg.Game run
INFORMATION: Player:CPU{health=7, mana=0/8, hand=[5], deck=[0, 1, 3, 3, 3, 4, 4, 6, 7]} wins the game!

Process finished with exit code 0
```

Player Input via System.in

AI beats me! Again! ☹

Thank you for your attention!

Recording of this talk at „Agile Saturday X“ in Tallinn, Estonia
https://vimeo.com/92886146

https://twitter.com/bkimminich
https://linkedin.com/in/bkimminich
https://google.com/+BjörnKimminich
http://slideshare.net/BjrnKimminich/

Credits

Background Image: Demonplay (Permission of use granted)

Other Images:

- Blizzard Entertainment (Permission of use requested)
- Wikipedia, Github, Agile Estonia
- Allmystery.de, Gueball.de, Natekohl.net, Scenicreflections.com